Head Motion Controlled Endoscopic Camera Manipulation using Inertial Motion Sensor and Mixed Reality Headset

Y.X. (Yoeko) Mak

MSc Report

Committee:
Prof.dr.ir. S. Stramigioli
Dr.ir. M. Abayazid
Dr.ir. H. Naghibi Beidokhti
Dr. C. Brune

December 2018

048RAM2018
Robotics and Mechatronics
EE-Math-CS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Summary

A minimally invasive surgery (MIS) procedure normally requires an assistant to hold and control the endoscope camera, such that the assistant will move the camera while the surgeon performs the surgical operation. A head motion controlled endoscope will allow the surgeon to control the camera directly.

During the master's assignment, a head motion controlled endoscopic camera system has been developed and evaluated. First, a system is built using an inertial measurement unit (IMU) attached to the surgeon's head as input and a standard display monitor as output, followed by using a Microsoft Hololens Mixed Reality headset as additional visual output. One challenge in realizing the system is to compensate for the rotation of the endoscope tip in order to have the visual output stable orientation-wise.

A pointing task experiment is conducted with help from clinicians and trained non-clinician participants to compare the human performance while using the head-controlled system versus manual usage of the flexible endoscope. The acceptability of the developed endoscope system is evaluated through a questionnaire.

Results from the pointing task experiments show that the developed head-controlled endoscope has significantly faster reaching time performance in the high index of difficulty (ID) task with clinician participants ($p = 0.0435$), and in both ID tasks with non-clinician participants ($p = 0.0290$ for ID= 2 and $p = 0.0351$ for ID= 3), against the manual usage of the flexible endoscope. Moreover, the questionnaire responses indicate that the head-scope system is acceptable among clinician and non-clinician participants. More importantly, the head-scope system offer several advantages with respect to the current practice, such as direct camera control by the surgeon, better ergonomics in the operating room (OR) due to fewer personnels, and reduced post-operative pain induced by the pressure from the rigid endoscope to the rib cage

Acknowledgements

First and foremost, I would like to thank my parents for their unrelenting support throughout my study here in University of Twente. Without their support, I would not be in this position, enjoying what I love to do, making things and tinkering with gadgets.

I would like to express gratitude to dr. ir. Momen Abayazid for the opportunity of doing this project, broadening my perspective into the clinical research field and being able to work with people, doctors, and surgeons in this master's assignment. He has been a caring mentor before being my supervisor, providing his guidance and advice during my learning process in research and writing this thesis. I would like to thank dr. ir. Hamid Naghibi Beidokhti for his support in research, providing feedback, and also regarding non-technical aspects throughout my thesis work. I would like to thank prof. dr. ir. Stefano Stramigioli for his feedback during this project and also for being the chairman of the graduation committee, and dr. Christoph Brune for being the external member. I want to thank our collaborator from Universitair Medisch Centrum Groningen (UMCG), notably Maurits Zegel and prof. dr. Massimo Mariani for their support during the experiment work with the clinicians.

I had a great time at RaM, working together with other master students that provide friendly but productive working environment in- and outside the lab. I want to thank all colleagues, technicians, and friends, those who provide support during my stay at RaM, also to fellow master students for the precious 'koffietijd' during my study: Jornt, Jeroen, Maryam, Toon, Rik, Koen, and Adrian.

I want to thank my beloved partner, Silke Tara, who helped me tremendously with day-to-day and mental support throughout my study. After waiting for so long, you can finally realize your dream: 'get a new apartment, adopt a dog, period'.

Finally, thank you as the reader of my thesis. If you read this report because you are using my code in RaM Gitlab, I hope it is tolerable, please enjoy my thesis.

Yoeko Xavier Mak
Enschede, 11th December, 2018

Contents

1 Introduction **1**
1.1 Context . . . 1
1.2 Problem statement . . . 2
1.3 Related works . . . 2
1.4 Research objectives . . . 4
1.5 Report outline . . . 4

2 Background Theory **6**
2.1 SimpleFlow: a dense optical flow algorithm . . . 6
2.2 Robot Operating System (ROS) . . . 8
2.3 Fitts' Law . . . 9

3 Head-scope System Modules **10**
3.1 Flexible endoscope tip actuation . . . 11
3.2 Image rotation compensation . . . 14

4 Software Development in Robot Operating System (ROS) Framework **16**
4.1 General system architecture using ROS . . . 16
4.2 Modules . . . 17

5 Experimental Setup **21**
5.1 System modules technical validation . . . 21
5.2 Pointing task experiment . . . 22

6 Results **26**
6.1 Technical validation of system modules . . . 26
6.2 Pointing task experiment . . . 28

7 Concluding Remarks **32**
7.1 Discussion . . . 32
7.2 Conclusion . . . 33
7.3 Recommendations for future work . . . 33

A Appendix 1: Endoscope Camera Calibration **35**
A.1 Calibration models . . . 35
A.2 Calibration tools . . . 36
A.3 Endoscope camera calibration using Kalibr . . . 37

B Appendix 2: Code Repository **39**

C Appendix 3: Head-scope pointing task experiment documents **40**

C.1 Instruction manual for participants 40

C.2 Consent and questionnaire form 42

Acronyms **46**

Bibliography **47**

1 Introduction

1.1 Context

Innovations in thoracic surgical techniques and endoscopic devices have enabled surgeons to perform less invasive treatments as opposed to conventional methods. One of such minimally invasive procedures is video-assisted thoracoscopic surgery (VATS). In VATS, an endoscopic camera is inserted into the patient's body via a small incision in-between the rib cage to give the surgeon visual feedback.

This procedure normally requires an assistant to hold and control the endoscope, such that the assistant will move the camera while the surgeon does the surgical operation (Figure 1.1). The surgeon will then instruct the camera assistant to move the camera to the desirable position. This communication line between the surgeon and camera assistant often becomes a hurdle in navigating and positioning the surgical tools appropriately.

Figure 1.1: Operating room and personnel setup during a VATS operation, from Agasthian (2013).

Offloading the endoscope camera control from the assistant to surgeon may offer a more straight-forward method of maneuvering the endoscope camera. Using a robotic system to maneuver the endoscopic camera can bring additional benefits compared to the current rigid endoscope system (Figure 1.2) used in thoracoscopy, such as better vision range and less pressure applied to the patient's rib cage during the surgical operation.

The development of robotic surgery systems also offers many solutions regarding space and ergonomy during VATS. Review article by Kalan et al. (2010) details the development of many surgical robotic systems, starting with *AESOP*, a voice-controlled laparoscopic camera holder developed in 1996, enabling surgeons to use their voice to control the camera.

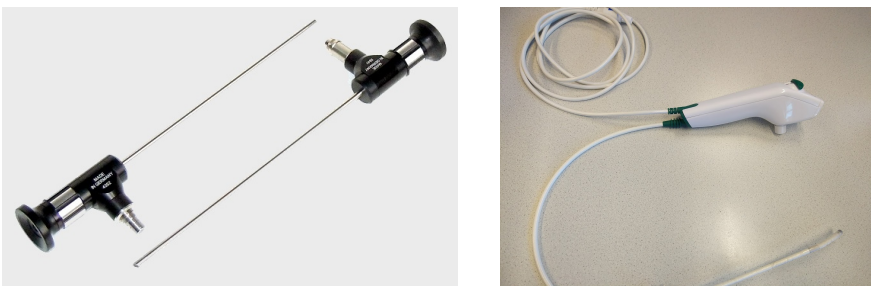


Figure 1.2: Rigid endoscope head (left image) versus flexible endoscope system (right image).

1.2 Problem statement

While minimally invasive surgery (MIS) is less-invasive and presents beneficial impact to the patient's recovery (compared to traditional open-surgery), this procedure has several drawbacks. We look into these drawbacks, especially those that are related to the visual-feedback system:

- **Control input** of the endoscopic camera, which is handled by the assistant, creates a limiting factor such that the camera control is dependent on the communication between surgeon and assistant. There are several solutions that enable the surgeon to control the camera directly (as we will discuss later in Section 1.3). This is quite tricky since both of the surgeon's hands are occupied with surgical tasks / other tools.
- **Space and ergonomic condition** during such minimally-invasive procedures are very limited since many people and tools need to work in a relatively small space. As the procedure requires several medical personnels to perform, the surgeon and assistants may not always have a direct / comfortable line of sight to the monitor, and the cramped work area contributes to added stress and fatigue, especially for long operations.
- **Maneuverability** of the endoscope inside of the patient's body is limited. The endoscope should not apply excessive force to the cavity wall or interfere with other instruments.
- **Added pain and post-operative recovery needed** due to the pressure applied to the patient's rib cage in surgery using the currently used rigid endoscope system. Most of the time when a surgical operation is performed in places hard to reach (with the rigid endoscope system), extra force is applied on the scope handle to accommodate for proper vision feedback. In extreme cases, based on feedback from the clinicians who participated in this project, the rigid-scope's shaft could get bent from the force applied during operation.

1.3 Related works

Since then, many different solutions have been proposed to control endoscopic cameras more intuitively. Different approaches can be classified based on the control input:

- hand controlled: single-handed controller approach (Rozeboom et al., 2014)
- foot controlled
- body controlled: *PMASS* (Martinez et al., 2009)
- voice controlled: *AESOP* (Mettler et al., 1998)
- gaze/eye-movement controlled (Noonan et al., 2010)
- head controlled (Reilink et al., 2010)
- self/image-guided (van der Stap et al., 2014, 2015)

These approaches are still active research topics, especially the image-guided approach. This has been a result of increasing GPU computing power and recent increase in popularity of vision-based tracking algorithms.

In this study, we specifically will look into the head-motion controlled endoscopic camera, where an inertial measurement unit (IMU) or any other positioning system is placed on the surgeon's head to control the camera movement. With a head controlled endoscopic camera, we aim to minimize the amount of fatigue or burden felt by the surgeon during operation.

One highly relevant work is presented by Reilink et al. (2010), supplemented by master's thesis of de Bruin (2010) that contains the design considerations, details a system-level implementation of a head-motion controlled flexible gastroscope, complete with a head mounted display (HMD) used as the display output. However, neither clinical evaluation or comparison against existing systems was performed in their work. Three control modes were tested in their work, position-control, rate-control, and a hybrid between the two. They concluded that the performance of the system using position and rate-control are better than the hybrid control scheme.

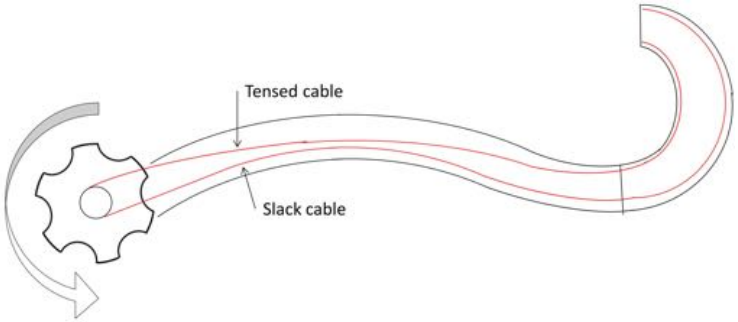


Figure 1.3: Tip deflection using two antagonistic cables running inside the endoscope, image from (Rozeboom, 2016, p.85).

The endoscope used in our project has one degree of freedom (DOF) actuation in deflection (Figure 1.3), therefore to enable 2 DOF movement, the endoscope (with the handle) has to be rotated around the cable axis. The image produced by the endoscope may rotate due to this rotation motion, creating a less-intuitive visual output, where up-direction in head movement does not correspond to up-direction in the output image. A paper titled Automatic Endoscopic Image Orientation Stabilisation with Ultra-low-latency was published by Van Ranst et al. (2016). The authors used the Kanade-Lukas-Tomasi algorithm (Tomasi and Kanade, 1991) to track the image rotation and then extract the corresponding rotation angle. The ultra-low-latency implementation was completely dependent on a ready-to-use system called NucleUS (developed by eSATURNUS, acquired by Sony in 2016). This low-latency pipeline was not expanded in detail.

With regard to the output device used for visual output display, a head mounted display can be used as opposed to a standard monitor to improve the ergonomics during surgery. HMD is already used in hospitals that use state-of-the art surgical equipment, coupled with high-definition 3D endoscope and the Da Vinci Surgical System (Kihara et al., 2012).

Another important factor that highly contributes to the surgeon's performance with the usage of a robotic flexible endoscope is the *motion-to-photon* latency, which is the time delay between the time when input motion is prescribed and the corresponding change can be seen at the visual display. A study on the effect of latency on steering tasks was published, which showed degradation in pointing & steering performance, specifically the movement time, starting at a latency of 86 ms (Friston et al., 2016). The mapping between motion reference input to the endoscope tip motion, also called the control-display gain, and its effect on pointing and steering tasks was thoroughly investigated in (Casiez et al., 2008).

Several preceding works have been done in this project by previous students. First, mechanical hardware to actuate the flexible endoscope has already been designed and built, and second, an image rotation compensation module has been made previously based on the Lucas-Kanade optical flow algorithm (Bouguet, 2001) using Open Source Computer Vision Library (OpenCV). This implementation of image rotation compensation was not very effective for use with the output image from the endoscope, due to poor image quality and lacking input preprocessing steps in the implementation.

1.4 Research objectives

The primary goal of this project is to develop a head-controlled endoscopic camera system, serving as a proof of concept that the head-scope system is feasible to be used for a minimally invasive surgery procedure. This means that the system has to be accurate, easy-to-use, and intuitive to the clinicians. The feasibility of the head-controlled endoscope system for use in MIS has to be tested in a clinical setting.

Consequently, a research question is proposed for this project:

To what extend, can a head-controlled endoscopic camera offer a solution to the camera control limitation in minimally invasive surgery (MIS)?

Sub-questions:

RO1 How should the head to endoscope tip orientation mapping be implemented to make the system more intuitive and easy-to-use?

RO2 How should the image rotation compensation for the visual feedback to the surgeon be implemented to minimize rotational motion of the output image?

RO3 How should the system be developed in terms of software to accommodate Hololens integration while keeping latency minimal?

RO4 How good is the human performance using this system, in terms of time and accuracy, compared to manual usage?

RO5 How acceptable is this system, as seen by the clinical end-user?

Regarding the head-scope system's development, an important part to be considered is to remove the rotation component in the motion of output image, such that directionality in the image is kept the same as the head movement. To achieve this, a dense optical flow algorithm is employed at uniformly-distributed pixels in the image, and the transformation between image frames can be estimated using the flow vectors. Ultimately, a Microsoft Hololens mixed-reality headset will be used to display the visual feedback from the endoscope, therefore the user does not have to fixate his eyes to the display monitor. The system then will be tested with the clinicians from Universitair Medisch Centrum Groningen (UMCG), to assess the performance of the head-scope system compared to manual usage of the flexible endoscope.

1.5 Report outline

This master's project report is outlined as follows:

This chapter (Chapter 1) presents a general description of the project, problem statement, and the research objectives.

Chapter 2 explains the underlying theory of the SimpleFlow algorithm used for the image rotation compensation module, and a brief explanation about the Robot Operating System (ROS) environment / terminologies.

In Chapter 3, the inner workings of each head-scope system's module are presented. Design consideration and details of each module are explained in this chapter (**RO1** & **RO2**).

Chapter 4 presents the software architecture of the head-scope system, built in ROS environment (**RO3**).

Chapter 5 presents the experimental setup used for the validation of the head-motion controlled endoscope camera system. The validation covers both technical validation of the system's modules and also pointing-task experiments done with the clinicians from UMCG.

Chapter 6 presents technical validation results, pointing task experiment results (**RO4**), and the clinician's questionnaire results (**RO5**). Finally, discussion, conclusion, and future recommendations are presented in Chapter 7.

2 Background Theory

Tools and algorithms used in this project will be explained in this chapter.

2.1 SimpleFlow: a dense optical flow algorithm

One challenge in realizing a head-controlled endoscope camera system is in delivering the right visual output to the system's user, specifically due to the endoscope tip rotation. Without an image rotation compensation scheme, directionality in the image plane will vary during movement, creating a non-intuitive behavior to the user.

Optical flow algorithm is used to detect the change between consecutive image frames, and the rotation is directly estimated in the image-domain. Other methods to estimate the image rotation are presented later in Chapter 3 (Section 3.2).

The optical flow algorithm needs to detect flow in the overall scene, not object-focused, for example, the algorithm must not be sensitive to the movement of surgical tools seen in front of the camera. Accuracy and computational cost are important metrics to be considered when selecting which optical flow algorithm to use. An image rotation compensation algorithm has been done previously using sparse Lukas-Kanade optical flow, however the accuracy of rotation estimate was unsatisfactory.

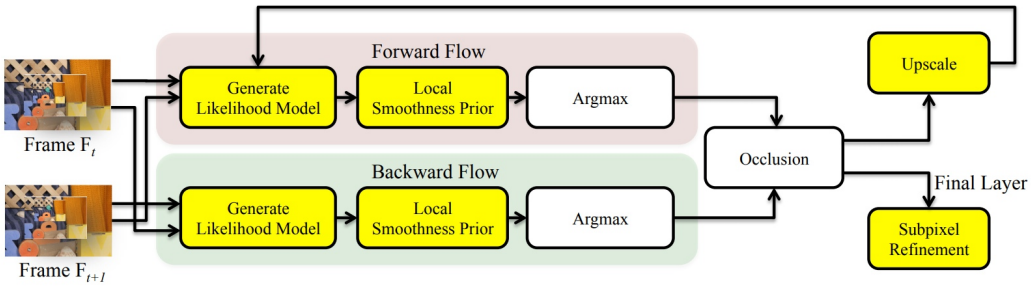
Several optical flow algorithms were considered as method to obtain flow vectors between consecutive frames. These algorithms are grouped into sparse and dense type according to the number of pixels considered as input. Sparse algorithms use specific points in the input frame (strong corners, highest gradient, etc.), while dense algorithms use all or majority of pixels from the input frame. The optical flow library in OpenCV offers ready-to-use application programming interface (API) for sparse Lukas-Kanade flow algorithm (Bouguet, 2001), and several dense algorithms such as: Farneback (Farnebäck, 2003), Brox (Brox et al., 2004), and SimpleFlow (Tao et al., 2012).

SimpleFlow is chosen due to several considerations: it has lower computational cost compared to other dense algorithm in OpenCV, it smooths out the flow vector over patches with similar color, and also has occlusion detection. SimpleFlow algorithm (Tao et al., 2012) used in the image rotation compensation module will be summarized in this section.

SimpleFlow is a dense optical algorithm that utilizes a color invariance assumption, meaning the same point / object in different image frame is assumed to have same color vector. The magnitude of this color vector difference is the main component of the cost function which the algorithm tries to minimize.

The process of SimpleFlow algorithm (with process block diagram presented in Figure 2.1), is as follows:

1. Define an error function based on the magnitude of color difference using initial guess of flow (u, v):
$$e(x, y, u, v) = \left\| F_k(x, y) - F_{k+1}(x + u, y + v) \right\|^2 . \tag{2.1}$$
2. Assume (u_0, v_0) at pixel (x_0, y_0) is a good explanation of the motion over surrounding pixels in patch $\mathcal{N}_0$. Smooth out the error function over patch $\mathcal{N}_0$ using a bilateral filter



Speed up factor over single threaded CPU implementation

	Generate	Smoothness	Upscale	Subpixel	Total
8 Threads CPU	4.42	3.18	4.78	5.06	4.62
8 Cores CPU	6.97	3.88	6.06	7.48	5.95
GPU	33.87	4.49	101.56	81.84	29.60

Figure 2.1: Illustrating different stages of the SimpleFlow algorithm pipeline. Stages highlighted in yellow are parallelized. [image from Tao et al. (2012)]

with weights based on color difference and pixel distance to (x_0, y_0):

$$\begin{aligned} E\left(x_0, y_0, u, v\right) &= \sum_{(x,y)\in\mathcal{N}_0} w_d\, w_c\, e(x, y, u, v)\,, \\ \text{with } w_d &= \exp\left(-\left\|\left(x_0, y_0\right) - (x, y)\right\|^2 / 2\sigma_d\right), \\ \text{and } w_c &= \exp\left(-\left\|F_k\left(x_0, y_0\right) - F_k(x, y)\right\|^2 / 2\sigma_c\right). \end{aligned} \tag{2.2}$$

3. Flow (u_0, v_0) can be computed by minimizing E:

$$(u_0, v_0) = \arg\min_{(u,v)\in\Omega} E\left(x_0, y_0, u, v\right), \tag{2.3}$$

where Ω is the set of possible (u, v) vectors considered from initial guess.

4. Occlusion detection is done by comparing the forward flow u_f, v_f (from F_k to F_{k+1}), with the backward flow u_b, v_b (from F_{k+1} to F_k). Ideally these two flows should be the opposite of each other. So when the value of $\|(u_\mathrm{f}, v_\mathrm{f}) - (-u_\mathrm{b}, -v_\mathrm{b})\|$ is above some threshold, then the pixel can be confirmed as occluded.

5. Step 1-4 is done at several image pyramid level, starting from coarsest to the original image resolution. The flow estimate from previous level is up-sampled using joint bilateral upsampling (Kopf et al., 2007) and used as initial guess for the next level.

6. At the final image pyramid level, the flow result is further regularized by applying bilateral filter with weights w_d, w_c (from Equation 2.2), and an extra weight w_r that represents the reliability of the flow estimate at (x, y):

$$w_\mathrm{r}(x, y) = \operatorname*{mean}_{(u,v)\in\Omega} e(x, y, u, v) - \min_{(u,v)\in\Omega} e(x, y, u, v) \tag{2.4}$$

SimpleFlow algorithm API is available in OpenCV `optflow` library, however only CPU implementation of this algorithm is available.

2.2 Robot Operating System (ROS)

For project that involves a system level development such as this, a framework that enables modular code structure is required. Other than modularity, the software needs to deliver low-latency communication between several devices (host computer, Hololens, and Arduino). ROS offers a solution for these necessities: modular software structure, easy implementation of communication between different systems, high code re-usability, and easy integration across different languages and environments.

> ROS is an open-source, meta-operating system for your robot. It provides the services you would expect from an operating system, including hardware abstraction, low-level device control, implementation of commonly-used functionality, message-passing between processes, and package management. It also provides tools and libraries for obtaining, building, writing, and running code across multiple computers. — *Willow Garage (2018, p.1)*

ROS is used as the development framework for this project as it has many features enabling easy communication between devices, and also the modular structure in software makes a good foundation for future addition / implementation. The general architecture of ROS is based on peer-to-peer approach where multiple processes can be run in multiple host machines and communicate with each other. The coordination of these communications are the main role of a ROS master node.

ROS is only available in Linux operating system, while the newer version (ROS2) supports Linux, Windows, and Mac OS X, but is currently under heavy development.

2.2.1 Nomenclature

Some ROS terminologies are used in later chapters, therefore those specific ROS-related terminologies will be clarified in this section. A more extensive explanation about ROS concepts are presented by Romero (2014).

node: A ROS node is an executable that performs computation in a ROS framework. Two main programming languages is used to develop a ROS node, Python and C++, and a wide selection of APIs are available for communication with different languages / environment such as C#, Lisp, Java, JavaScript (specifically Node.js), Lua, R, Ruby, and several others.

message: ROS messages are used by ROS nodes to communicate with each other. The structure of a ROS message is built using primitive data types such as int, float, double, etc.

topic: ROS topics are the communication channels where nodes can exchange messages. Nodes that interested in particular messages *subscribe* to the relevant topic, while nodes that generate data *publish* to the relevant topic. A topic is defined in a string format using Graph Resource Names, structured in a hierarchical namespace separated by forward slash character '/'. This hierarchical naming structure is also used to used to define nodes, services, and parameters.

service: ROS service is used for synchronous communication between nodes. A node can provide a service that can be called with a 'request' message from another node, which then sends a 'response' in return. The syntax of these request and response messages are identical to the messages used with ROS topics.

parameter: A parameter server is a feature from ROS that stores values used by the nodes. ROS nodes use the parameter server to store and retrieve parameters at runtime. These parameters can be loaded at initialization through a launch file using a configuration file written in `.yaml` format. This YAML [abbreviated from YAML Ain't Markup Language (YAML) (YAML.org, 2006)] configuration file can be treated as list of used settings, as it is designed to be a human friendly data serialization standard for any programming languages.

package: ROS package is a collection of nodes & services along with the (custom) messages and parameters used by each node. ROS packaging system is structured in a catkin workspace. Catkin is the official build system of ROS, which combines CMake macros and Python scripts to provide some functionality on top of CMake's standard usage. The ROS package system enables a high-level integration of the entire ecosystem, supported by many features such as: a shared repository / central database of ROS packages, wiki page, documentation, and independence w.r.t. Linux distributions.

2.3 Fitts' Law

In order to validate the performance of head controlled endoscope system, evaluation metric such as accuracy has to be defined beforehand. Fitts' law expresses the relation between motion time and difficulty of the pointing task (Section 2.3). For task where higher pointing accuracy is needed (smaller target), higher motion time is expected.

Fitts' law has been shown to work under range of conditions: with different limbs [hands, feet, head-mounted sights (So and Griffin, 2000), and eye gaze (Zhang and MacKenzie, 2007)], different input devices (MacKenzie et al., 1991), and different user populations.

Fitts' law is a predictive model of human movement, which says that the motion time in a reaching task is linearly proportional to the index of difficulty (ID), where ID is the logarithm of ratio between target distance to the width. The most frequently used formulation of the index of difficulty is called the Shannon formulation (Soukoreff and MacKenzie, 2004):

$$t_{\text{motion}} = a + b \cdot \text{ID}, \tag{2.5}$$

$$\text{ID} = \log_2\left(\frac{d}{w} + 1\right), \tag{2.6}$$

where d is the distance to the target's center, w is the target width, and the +1 at the end is added to ensure ID stays positive.

A validation setup can be constructed to assess human performance in using the head-scope system, by setting the target size based on this definition of index of difficulty and the distance to the target. We can set up a comparison trial where the difficulty of pointing task is controlled / kept constant, and the motion time is measured for the usage of each system. This way, the difficulty of the pointing task validation experiment is controlled, and higher motion time should be expected with task that requires more accuracy (higher ID, smaller target).

3 Head-scope System Modules

The primary goal of this project is to develop a head-controlled endoscopic camera system, serving as a proof of concept that the head-scope system is feasible for use in minimally invasive surgery (MIS).

One important part of the head-scope system is to remove the rotation component in the motion of the output image, to make it intuitive to the user. Finally, Microsoft Hololens mixed-reality headset will be used to display the visual output, such that the user does not have to fixate his / her eyes to the display monitor.

The users of this head-scope system still need to be able to move their head freely when they are not controlling the endoscope tip movement. Therefore, some control trigger mechanism is required. A footswitch (Figure 3.1) is used to enable / disable the control of the endoscope camera, enabling the surgeon to move his head freely when the footswitch is not pressed. The footswitch also serves a second function, which is to rotate the image horizon manually. This feature is requested by the clinicians, as in the operation procedure, the patient's torso is not always horizontal with respect to the ground, and manual horizon adjustment is needed at the start. Practically, this is achieved by differentiating short-press (single click) for manual image rotation versus long-press to enable the endoscope control.

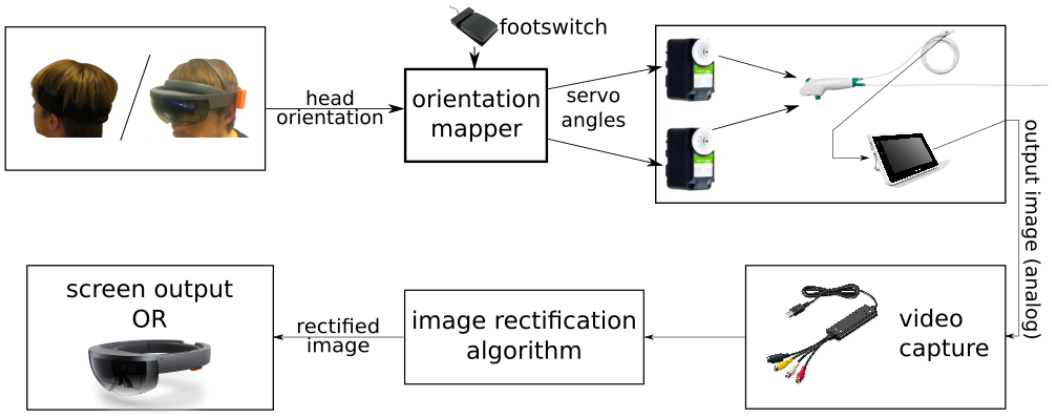


Figure 3.1: Block diagram showing general overview of process in the head-controlled endoscope system. Top-left block: head orientation is measured with a tracker attached to a headband or to the Hololens headset.

The image output from the endoscope system will be rectified, meaning that orientation in the image plane will be kept constant (as discussed earlier in Section 1.4). This step is necessary in order to have a system that is intuitive and easy to learn. The visual feedback given to the surgeon is done by simply displaying the rectified image on a display monitor or can be displayed on a transparent head mounted display, in this case we are using Microsoft Hololens mixed-reality headset. The Hololens implementation is not tested in a clinical setting due to project schedule and planning with the clinicians who participated in this project.

The head-scope system can be separated in several modules based on its function:

1. Head orientation data measurement.
2. Servo controller (orientation mapper), which involves kinematic mapping from sensor frame to the camera frame, listening to the incoming footswitch signal, and sending joint position to the servos.

3. Hardware: endoscope motorized gripper, Ambu flexible scope with accompanying monitor, and the frame-grabber to read the images.

4. Image rotation compensation module that stabilizes the output image rotation.

5. Visualization, either displaying it on a monitor, or sending a compressed output image to the Hololens device.

3.1 Flexible endoscope tip actuation

3.1.1 Head orientation measurement

Several measurement methods can be used to obtain the head orientation information:

- By attaching inertial sensors to the head of the user, as demonstrated by Reilink et al. (2010).
- Using visual odometry algorithm from cameras placed on user's head, such as the built-in positioning system of Microsoft Hololens mixed reality headset (in conjunction with inertial sensors).
- Using a head-pose estimation algorithm (from a monocular camera input), such as the algorithm developed by Lemaignan et al. (2016), where face-features are detected and then matched to a 3D face model, hence face orientation is obtained.
- Using optical marker placed on the head of the user, in conjunction with infrared (IR) cameras to triangulate the marker positions. A commercial solution for this approach is widely available, however is considerably costly.
- Using optical based triangulation between optical sensors placed on the head of the user and a static reference point, also called the 'lighthouse tracking' system. This method is used by HTC Vive as localization method for their virtual reality (VR) system.

For this project, head orientation measurement using inertial sensors is chosen, not only because it is already available for the project, but also due to the robustness and very little computational power is required. Optical-based approach requires uninterrupted line-of-sight to the camera / reference optical source, which is not suitable in an OR environment.

The IMU used in this system is a wireless Xsens MTw Awinda sensor, which has an accelerometer, gyroscope, and magnetometer in a single sensor assembly (Figure 3.2). The fusion of accelerometer, gyroscope, and magnetometer is done on-board the IMU using their proprietary Xsens Kalman Filter (XKF) algorithm, and the output orientation is accurate up to 1.5° RMS (dynamic). Further detail on the performance and characteristics of this IMU is presented in the white paper by Paulich et al. (2018).

Figure 3.2: Xsens MTw Awinda wireless IMU (left) and older wired 3rd-generation Xsens MTi IMU (right).

Earlier in the project, a third-generation Xsens MTi wired IMU was used. The new wireless IMU provides more flexibility and movement freedom since user's head is not tethered. One draw-

back is that the wireless IMU introduces 30 ms additional latency due to the wireless transport (from the datasheet of Xsens MTw Awinda).

3.1.2 Motorized endoscope gripper and video capture device

Figure 3.3: Motorized gripper with 2 servomotors to actuate the tip rotation and deflection. Actuation mechanism with Ambu endoscope attached (left) and without the endoscope (center), and the handle of the Ambu aScope 3 Regular (right).

The endoscope gripper hardware has already been made by previous student who worked on this project. One servomotor is used to rotate the handle around the cable axis, and another servomotor is used to actuate the thumb lever mechanism to control the deflection of endoscope tip (Figure 3.3).

In a classical control terms, what we are doing is only mapping the kinematics of the head to the endoscope tip, therefore we have no control over the force applied by the motor. The servomotors take position as input, and have their own feedback loop (PID control) on-board. This position-control mechanism is satisfactory since we will not implement haptic feedback in the current scope of the project.

As mentioned briefly in Chapter 1, the endoscope used in this project is a one degree of freedom flexible endoscope (Ambu aScope 3 Regular). The endoscope comes with a monitor and uses a proprietary connector to connect the endoscope to monitor. The transport of image signal from the endoscope to the monitor also uses some proprietary analog image encoding. Fortunately, image from the monitor can be exported through an RCA port, and then grabbed using an analog video capture card. This hardware pipeline is clearly not ideal as it introduces significant amount of latency due to the unnecessary image processing done inside the Ambu monitor.

For capturing the analog image output from the Ambu monitor, a Terratec G1-USB frame grabber is used to capture the endoscope image into a digital format. The resulting output is a 25 frame per second (fps) video stream with 6-7 frame delay from the endoscope to PC screen.

3.1.3 Orientation mapper / servo controller

Tip orientation to joint space mapping

First, the mapping from head orientation to joint position needs to be determined. Since the joint position is dependent on orientation only and translation component of the camera frame is not important, quaternion algebra is used for simplicity of programming. Let $\mathbf{q}$ to be the transformation from straight to current tip orientation, and endoscope straight position points along x-axis, such that bearing vector $\mathbf{r}_{\text{straight}} = \begin{bmatrix}1 & 0 & 0\end{bmatrix}^\mathsf{T}$. Then current bearing vector $\mathbf{r}$ can be calculated using vector rotation by a quaternion:

$$\begin{bmatrix}0\\ \mathbf{r}\end{bmatrix} = \mathbf{q} \otimes \begin{bmatrix}0\\ \mathbf{r}_{\text{straight}}\end{bmatrix} \otimes \mathbf{q}^*, \tag{3.1}$$

where $\otimes$ represents the multiplication operation between two quaternions and $(\bullet)^*$ denotes the quaternion conjugate.

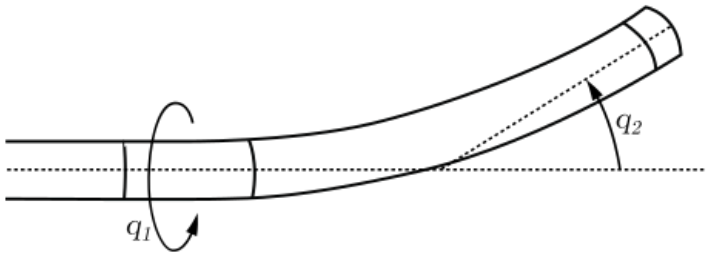


Figure 3.4: An illustration of the endoscope tip, showing the deflection and rotation axes actuated by the servo.

Then the joint position can be calculated,

$$q_1 = \text{atan2}\left(-r_y\,,\, r_z\right) \tag{3.2}$$

$$q_2 = \text{atan2}\left(\sqrt{r_y^2 + r_z^2}\,,\, r_x\right) \tag{3.3}$$

where r_x, r_y, r_z are the x-,y-, and z-component of $\mathbf{r}$.

Head to endoscope tip orientation mapping

One problem that might arise when the visual feedback to the user is displayed on a monitor, is the range of human head motion itself, since the range of endoscope tip bending is bigger than the range of human's neck movement. One solution is to use velocity based control, meaning the user controls the angular velocity instead of the orientation. However, from practical standpoint, this will introduce less intuitive control since humans are not used to control the rate of change of vision orientation using their head. Based from their experience in daily life, when a person move their head 90° to the right, they would expect to see things to their right.

We want to leverage the function of the footswitch, not only to enable the endoscope motion, but also to create a mouse-like functionality such that when the footswitch is pressed, the orientation will continue from the last orientation when the footswitch was released.

The orientation output from the IMU is given as current IMU orientation with respect to some earth magnetic field orientation $\mathbf{q}_e^{\text{imu}}$. This IMU orientation can be transformed to head frame (depending how the IMU is orientated on the headband):

$$\mathbf{q}_e^h = \mathbf{q}_{\text{imu}}^h \otimes \mathbf{q}_e^{\text{imu}}. \tag{3.4}$$

The transformation from IMU frame to head frame is $\mathbf{q}_{\text{imu}}^h = \begin{bmatrix}0.5 & 0.5 & 0.5 & 0.5\end{bmatrix}^\mathsf{T}$ when the IMU is placed on the headband (with up-right logo orientation) on the left temple of the user's head.

Then, the transformation for current footswitch 'session' can be derived as the orientation difference between current head frame and head frame at the time footswitch was pressed:

$$\mathbf{q}_{h_{\text{pressed}}}^{h_{\text{current}}} = \mathbf{q}_e^{h_{\text{current}}} \otimes \left(\mathbf{q}_e^{h_{\text{pressed}}}\right)^*. \tag{3.5}$$

The current orientation with respect to endoscope reference position can be derived using 'chain-rule' of all previous sessions multiplied by endoscope initial orientation $\mathbf{q}_{\text{ref}}^{\text{init}}$,

$$\mathbf{q} = \left(\mathbf{q}_{h_{\text{pressed,i}}}^{h_{\text{current}}} \otimes \mathbf{q}_{h_{\text{pressed},i-1}}^{h_{\text{released},i-1}} \otimes \mathbf{q}_{h_{\text{pressed},i-2}}^{h_{\text{released},i-2}} \otimes \ldots \otimes \mathbf{q}_{h_{\text{init}}}^{h_{\text{released},1}}\right) \otimes \mathbf{q}_{\text{ref}}^{\text{init}} \tag{3.6}$$

as head orientation frame at the time footswitch has just been pressed in session i is equal to the head frame at time the footswitch was released in session $i-1$ ($\Psi^{h_{\text{pressed,i}}} = \Psi^{h_{\text{released,i-1}}}$). The

initial endoscope tip orientation is set on 60° deflection from straight configuration to avoid the singularity problem.

Programmatically, Equation 3.6 can by simplified by saving the tip orientation at the time the footswitch was released in the previous session.

$$\mathbf{q}(t_{\text{current}}) = \mathbf{q}_{h_{\text{pressed}}}^{h_{\text{current}}} \otimes \mathbf{q}(t_{\text{prev-release}}) \tag{3.7}$$

Control-display gain

An important feature that is tightly related to the mouse-like behavior is the 'mouse sensitivity', or the control-display (CD) gain. It is the scaling between control input to the movement the user sees in the monitor / Hololens.

This behavior can be achieved using spherical linear interpolation (SLERP) (Shoemake, 1985, p.248) between identity quaternion $\mathbf{q}_I = \begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix}^\mathsf{T}$ and quaternion for the current session $\mathbf{q}_{h_{\text{pressed}}}^{h_{\text{current}}}$ (Equation 3.8).

$$\mathbf{q}_{h_{\text{pressed}}}^{h_{\text{current}}\prime} = \mathbf{q}_I \otimes \left(\mathbf{q}_I^* \otimes \mathbf{q}_{h_{\text{pressed}}}^{h_{\text{current}}}\right)^u, \qquad 0 < u < 1 \tag{3.8}$$

3.2 Image rotation compensation

There are 3 separate ways to obtain the image rotation data:

1. Using servo position information to infer the rotation in image plane (kinematics).
2. Using computer vision algorithm done in the image itself.
3. Using a motion tracker attached to the endoscope tip.

We will explore option 1 and 2 as no additional hardware/module is needed to implement these methods. While option 3 is used to verify the accuracy of the image rotation compensation module (more on Chapter 5).

3.2.1 Compensation using servo position information

The image rotation θ can be obtained by projecting the rotation q_1 into the image plane (see Figure 3.4).

$$\theta = q_1 \cos(q_2) \tag{3.9}$$

Afterwards, the image rotation compensation itself is trivial, by using the rotation θ to rotate the image.

$$\mathbf{R} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \qquad \text{and} \qquad \mathbf{t} = \mathbf{R}\begin{bmatrix} -\frac{w}{2} \\ -\frac{h}{2} \end{bmatrix} + \begin{bmatrix} \frac{w}{2} \\ \frac{h}{2} \end{bmatrix} \tag{3.10}$$

The translational term $\mathbf{t}$ exist as the rotation needs to be done around the center of the image. Image coordinate $(0,0)$ is defined as the top-left most pixel in the image, w and h are the width and height of the image, respectively.

3.2.2 Compensation using rotation estimated in image domain

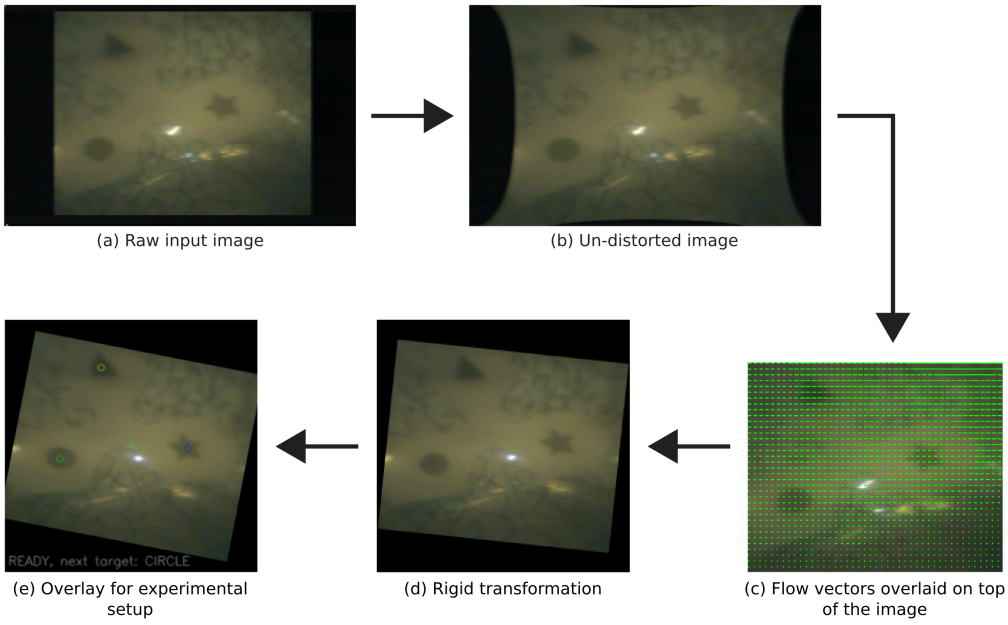
Estimating the image rotation directly using the image has advantages and disadvantages compared to using kinematics. One advantage is that the image compensation module is fully decoupled from everything else, therefore the compensation performance does not suffer from actuation non-linearities such as cable compliance, backlash, etc. The main disadvantage is that computer vision algorithm is typically costly in terms of computational power. Most dense optical flow algorithms run on graphics processing unit (GPU) due to the enormous computation requirement and the computation becomes parallelized. The image rotation obtained

using optical flow algorithms also suffers from drift due to the fact that the estimation error on the flow vector gets compounded over time.

The steps for rectifying the image using SimpleFlow algorithm (Section 2.1) is presented in Figure 3.5.

The input image needs to be pre-processed to remove image distortion, such that a straight line in the real world would appear straight in the image. The un-distortion step is done using a pinhole camera with radial-tangential distortion model (Appendix A). The endoscope camera is first calibrated to obtain the camera parameters and the distortion coefficients, then the image is un-distorted using Equation A.2. The resulting image is then cropped to remove high amount of warping near the border.

The image (size 220×180 px) is then down-sampled such that the new image is three times smaller in width and height compared to the original. This down-sampling is done because of the limitation in computational power, as the implementation we are currently using is CPU-based with multi-threads.



(a) Raw input image (b) Un-distorted image (c) Flow vectors overlaid on top of the image (d) Rigid transformation (e) Overlay for experimental setup

Figure 3.5: Visualization of image processing steps: (a) raw input image, (b) image after un-distortion step, (c) the un-distorted image is cropped, down-sampled, and then fed into the SimpleFlow optical flow algorithm; note that the down-sampling scale is increased to five for visual clarity in the image above, (d) the cropped undistorted image is rotated using the estimated angle from the flow vectors and then black border is added, (e) overlay used in experiment is then added afterwards (will be expanded in Section 5.2).

The output of the SimpleFlow algorithm are 2D flow vectors for each pixel in the input image. Using these flow vectors, we can estimate the rigid transformation between frames using Equation 3.11.

$$\begin{bmatrix} \cos(\theta)s & -\sin(\theta)s & t_x \\ \sin(\theta)s & \cos(\theta)s & t_y \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x+u \\ y+v \\ 1 \end{bmatrix}, \tag{3.11}$$

where (u, v) is the flow vector at image coordinate (x, y), and s is the scaling factor.

Practically, this transformation estimation is done using `estimateAffinePartial2D()` from OpenCV library, as this function also uses RANSAC (Fischler and Bolles, 1981) to remove outliers. The image is rectified using the same method as previous section (Equation 3.10), discarding the scale s and translation terms (t_x, t_y) from Equation 3.11.

4 Software Development in Robot Operating System (ROS) Framework

Robot Operating System (ROS) is used as the development framework due to its modular architecture and simple communication set-up across different devices.

4.1 General system architecture using ROS

The main requirements in term of software are manageable latency and modularity in structure. For this setup, most of the latency introduced by the image capture device, as the incoming analog video output has to be converted into digital video format for further processing. Pipeline to the Hololens headset also add some amount of delay as the connection to Hololens embedded platform can only be done through 2.4 GHz wireless channel.

In term of modularity, ROS framework creates a solid foundation for future software addition and increases code re-usablity. For example if the head orientation measurement method is changed in the future, for example using the Hololens built-in positioning system, current IMU reader package can be substituted by a orientation listener. The rest of the system does not need to change since the message transport in ROS is standardized across platforms.

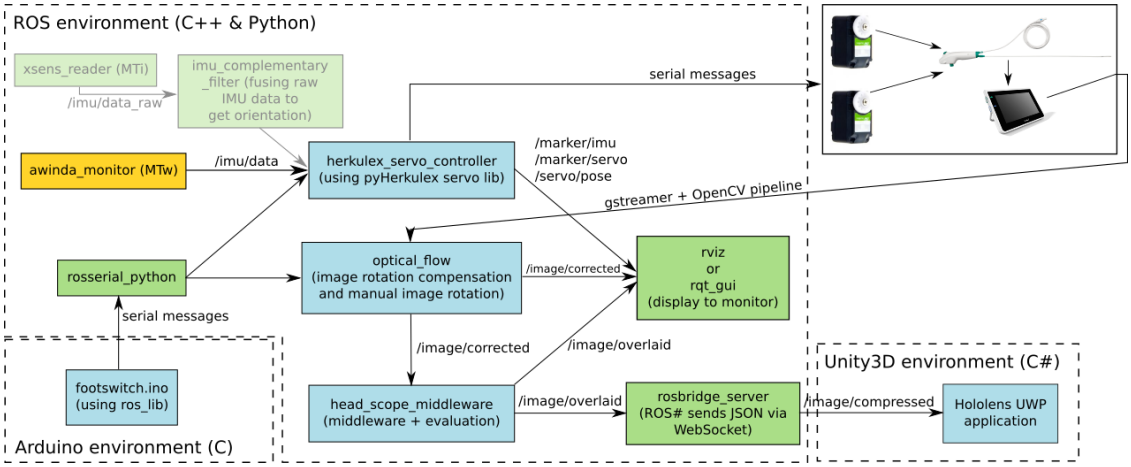


Figure 4.1: General software architecture at package level in ROS, color-coded in green: available / ready-to-use package, yellow: some modification on top of existing package, and blue: custom package / developed from scratch. Faded out packages are used for the old Xsens MTi wired IMU sensor.

Methods presented previously in Chapter 3 are realized in two separate ROS packages. `herkulex_servo_controller` deals with mapping and actuation of the endoscope gripper, and `optical_flow` deals with image rotation compensation (including compensation using servo kinematics). A middleware package is developed for a straightforward launch of the whole system, and it also serves as the software setup for the pointing-task experiment (expanded more in Section 5.2). A complete overview of the interconnection between packages are presented in Figure 4.1.

In addition to those three packages, an already available ROS package is used to read IMU orientation (`awinda_monitor`). An Arduino code is developed for the footswitch implementation. Furthermore, an Universal Windows Platform (UWP) application is developed in Unity, serving as a listener to the ROS message containing compressed output image.

4.2 Modules

Overview of networked nodes, topics, and message rates are presented in Figure 4.3. All packages used in this project is developed and test on ROS Kinetic in Ubuntu 16.04.

4.2.1 Sensor data reader

`/xsens_awinda_reader` node (awinda_monitor package)
Input: serial messages (Xsens protocol).
Output: `/imu/data` topic containing orientation quaternion of the IMU.

The `awinda_monitor` package is used to retrieve the IMU orientation data into ROS environment. This ROS package is based on the code made by Raffa87 (2017), contains C++ wrapper to the XDA library supplied by Xsens to read the data from MTw Awinda dongle. The package is modified to work on newer ROS version (ROS Kinetic).

4.2.2 Footswitch clutch

`/rosserial_footswitch` node (rosserial_python package)
Output: `/footswitch/enable_control`, boolean topic indicates footswitch is pressed,
`/footswitch/enable_manual_img_rot`, boolean topic indicates single click was performed (<250 ms).

The footswitch produces digital signal, which is read by an Arduino, sending two boolean messages directly to ROS environment using `ros_lib` library in Arduino IDE. `rosserial_python` package listens for the serial messages sent by the Arduino and translates them into ROS messages. One topic is dedicated for enabling endoscope tip control, and the other is to activate manual image rotation through a single-click input. A footswitch input is registered as a single-click when it is below 250 ms of press time.

The enable control message does not wait until end of the click. This is a design choice to make the footswitch feels more responsive. So if the user wants to change the image horizon at the start, the tip control will be triggered for that small click duration.

4.2.3 Endoscope tip controller / orientation mapper

`/servo_controller` node (herkulex_servo_controller package)
Input: `/imu/data`,
`/footswitch/enable_control`,
`/servo/position` (0-1023) for each rotation and deflection servo.
Output: `/servo/cd_gain`,
`/imu/marker` (for visualization in `rviz`).

`/servo_marker_publisher` node (herkulex_servo_controller package)
Input: serial messages from servomotors.
Output: `/servo/position`,
`/servo/pose`,
`/servo/image_rot_angle`,
`/servo/marker`.

This servo controller package consists of 2 nodes running asynchronously. `/servo_controller` functions as the orientation mapper, running the algorithm described in Section 3.1.3, and `/servo_marker_publisher` gets the servo position information for image rotation compensation, control, and visualization purposes.

4.2.4 Image rotation compensator

`/simple_flow_node` (awinda_monitor package)
Input: `/image/input` if running offline / without capture device, otherwise reads directly from capture device memory.
Output: `/image/rectified`,
`/image/flow`,
`/image/undistorted`,
`/image/input` when running using capture device.

The `/simple_flow_node` handles the image rotation compensation for both using optical flow and servo position (Section 3.2), and also running the frame-grabbing loop using OpenCV `VideoCapture()` class.

Other than image rotation compensation, feature such as manual image rotation to correct the image horizon is added per request by the participating clinicians. This is done simply by taking the roll angle of the IMU (head tilt) and use it in a rate-controller of the manual image rotation, which is activated by single-click of the footswitch.

The nodes in this package is developed in C++ (as opposed to Python with other packages in this project) for performance benefits, wider OpenCV's API options, and finer control over memory allocation. Note that the input image is downscaled by 3 in width and height before fed into the SimpleFlow algorithm due to limited computational resource. The accuracy performance and the loop-time of this module are presented in Section 6.1.2.

4.2.5 Experimental setup / overlay

`/overlay` node (head_scope_middleware package)
Input: `/image/rectified`,
`/servo/cd_gain`,
`/footswitch/enable_control`.
Output: `/image/overlaid`.

This package serves as a middleware and also implement features needed for the experimental setup (will be expanded in Chapter 5). This package serves several purposes:

1. as a middleware to launch the experiment routine for both the head-scope and manual usage experiment.
2. implements a blob detector using `cv::SimpleBlobDetector()` to detect pointing target centroid and assign the target diameter.
3. measure and record experiment's measurement data, such as measuring the motion time, recording target size / distance, etc.

The blob detector algorithm in OpenCV will first thresholds the image into binary values, then extracts connected components from the binary image and calculate their centers. Several basic shapes are used as pointing targets: circle, triangle, and star shape. Classification of the target can be done by tuning the circularity (area versus contour length ratio) and the convexity of the detected blobs.

4.2.6 Hololens integration using `rosbridge_suite`

`/rosbridge_websocket` node (rosbridge_server package)
Input: `/image/overlaid_proc/compressed`.
Output: *JSON* containing compressed overlaid image (*.jpeg*).

Before the image can be send to the Hololens, first it has to be compressed to lower the bitrate of data transmission. Using the `image_pipeline` package that is already implemented

in ROS, the overlaid image can be easily compressed and republished into different topic with *.jpeg* image format.

Afterwards, the compressed image is converted into JavaScript Object Notation (JSON) using `rosbridge_suite` package and `rosbridge_server` provides the transport layer using WebSocket. This is especially useful as WebSocket has the advantage of low-latency in video streaming delivery through the use of persistent connection (Wang et al., 2013).

Hololens app development in Unity

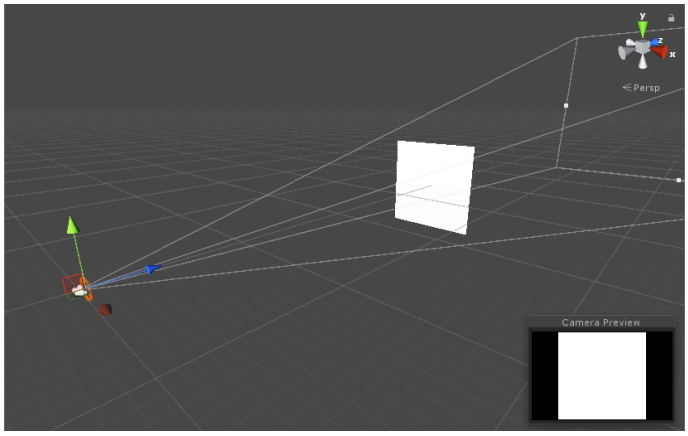


Figure 4.2: Visual output plane is placed 0.3 m in front of the Hololens camera position in Unity 3D development environment.

A more challenging engineering problem lies on re-interpreting the `rosbridge` protocol from Unity side and creating a Hololens UWP application with it. UWP is a type of application that is built in *.NET* framework (specific to Windows).

In the end, the currently functioning Hololens UWP app is developed based on a fork of *ROS#* library by Whitney (2018). *ROS#* is a set of open source software libraries and tools in C# for communicating with ROS from .NET applications, in particular Unity. This library will set up the communication between the `rosbridge_server` running in ROS / Linux with the UWP app running in the Hololens using JSON.

After the communication pipeline is sorted, the only task left is developing a 3D scene in Unity: a plane to display the output image that follows / faces the user however the camera is moved. The size of the image plane from the user's perspective is shown in bottom right corner of Figure 4.2.

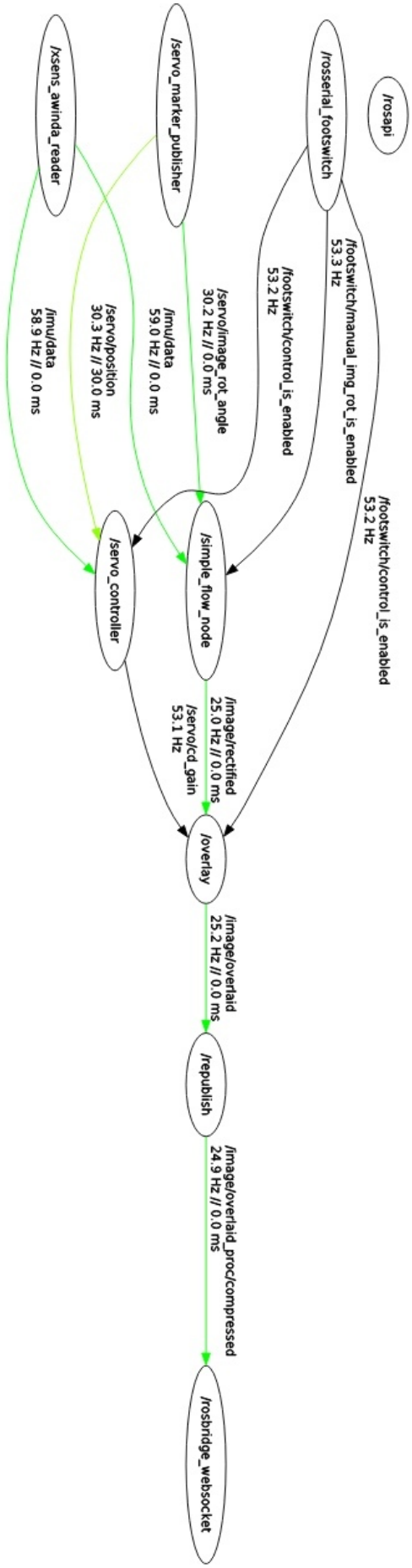


Figure 4.3: ROS nodes and topics running during head-scope usage with Hololens headset

5 Experimental Setup

In this chapter, the measurement setup for technical validation and experimental setup for the pointing task experiments are presented. Tip movement calibration, image compensation module validation, and image latency measurement are done to answer research objective **RO1** - **RO3**, while pointing task experiments using the developed head-scope system are done to answer research objective **RO4** - **RO5** (research objectives are presented in Section 1.4).

5.1 System modules technical validation

5.1.1 Tip movement calibration

Tip movement calibration is done to set the mapping between servo angle input to the real endoscope tip deflection angle. This is especially important since the transfer from the thumb lever to the tip deflection is not 1:1, unlike the rotational axis. Some non-linear behavior might also be present due to the Bowden cable mechanism used to actuate the endoscope tip.

Figure 5.1: Small 5 DOF electromagnetic tracker is placed on the endoscope tip (left) to obtain camera pose. NDI tabletop system under appropriate operating condition (right) has 0.5° orientation and 1.2 mm position root mean square error (RMSE).

An electromagnetic (EM) tracker (NDI Aurora 5DOF sensor) is placed on the tip of the endoscope, coupled with the field generator NDI tabletop system (Figure 5.1). The parameter to be measured in this setup is the tip deflection angle q_2, while a prescribed motion input is given to the servomotor. The servomotor will move the endoscope tip back and forth 8 times at 30°/s between the endoscope's maximum deflection angles.

5.1.2 Image compensation accuracy

The accuracy of the image rotation compensation algorithm can also be measured using the same setup as in the previous section, since the image rotation can be obtained by measuring the rotation normal to the camera plane.

Two estimated image rotation angles based on image rotation estimation methods from Section 3.2 will be compared against the EM tracker measurements used as ground-truth. The NDI Aurora tabletop system has 0.5° orientation RMSE in the operating conditions specified in the manual, which is far smaller than the expected uncertainties of the tip actuation and image rotation compensation error. Head motion is used as the motion input source for this validation procedure, where the speed and range of movement corresponds to simulated practical use-case of the system.

5.1.3 Latency measurement

Knowing the amount of delay in the output image is important as human performance in using the system will worsen with the amount of latency in the feedback loop (Friston et al., 2016). Image latency will be measured at several points in the image pipeline: Ambu monitor display, captured input image, post-processed output image, and at the Hololens display. Processing time of the image rotation compensation node will also be logged in the software.

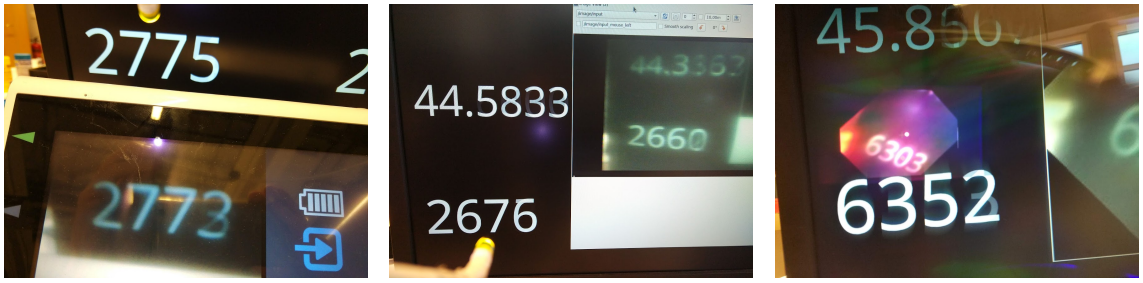


Figure 5.2: Frame counter at 60 fps and the corresponding image output at Ambu monitor (left), display monitor (center), and Hololens headset (right).

The image latency measurements are done by pointing the endoscope camera to a 60 fps frame counter. Then, the frame counter and the output endoscope image will be captured / snapshotted using a camera with fast shutter-speed (Figure 5.2). While this latency measurement method is straight-forward, it has low temporal resolution ($1/60\,\text{s} \approx 16.7\,\text{ms}$), therefore multiple snapshots ($N \geq 30$) are taken per point in the image pipeline, which will be averaged afterwards.

5.2 Pointing task experiment

The purpose of the pointing task experiment is to assess human performance quantitatively in using the developed head-controlled endoscope system. The measured performance metric is the reaching time, which is the time needed by the user to point the center of the endoscope image to a target. Accuracy requirement is prescribed into the task by changing the target diameter.

5.2.1 Design

The measured variable is reaching time, using 3 categorical independent variables, which are the usage of different systems: manual use of the flexible endoscope, head-controlled endoscope, and head-controlled endoscope with Hololens as visual output. Statistical analysis is performed to test for difference in means of reaching time with manual and head-controlled scope.

The pointing task experiment is conducted at two different difficulty settings, which correspond to the target size in the pointing task. ID of 2 and 3 (background theory presented in Section 2.3) are selected to simulate low and high accuracy use-cases in practical use. By using ID to define the target size, we aim to achieve 2 things: more controlled difficulty level for the pointing task experiment, and a relation between time and accuracy in context of the experiment. We prescribe different accuracy requirements into the experiment and the measured reaching time should increase linearly with ID.

5.2.2 Participants

The pointing task experiment is done on 2 separate occasions with 2 different sets of participants. The first set of participants consists of 4 clinicians from the cardio-thoracic department in UMCG, of which 3 are surgeons and one surgeon-in-training (males, age between 30-57).

The second set of participants have intermediate level endoscopic skill (non-clinicians, trained using laparoscope or other endoscope). A total of 8 trained non-clinician participants (6 males and 2 females, age between 23-33) participated in this experiment, 6 of which are senior Technical Medicine master students[1] who had completed a training in clinical insertions / endoscopy, and 2 are lecturers for an endoscopic training course at the University of Twente.

5.2.3 Procedure

During the test, the endoscope camera is placed on the inside of a spherical phantom (Figure 5.3). The endoscope is fixed in the z-direction (depth), for both head-scope usage and manual usage. Experiment participants will see an overlaid image stream from the endoscope camera on a screen (Figure 5.3, center and right image). The participants are tasked to move the center green cross-hair to the inside of a red circle overlaying the target. The target point is considered 'reached' when the center crosshair is inside of the red circle for at least 0.4 second.

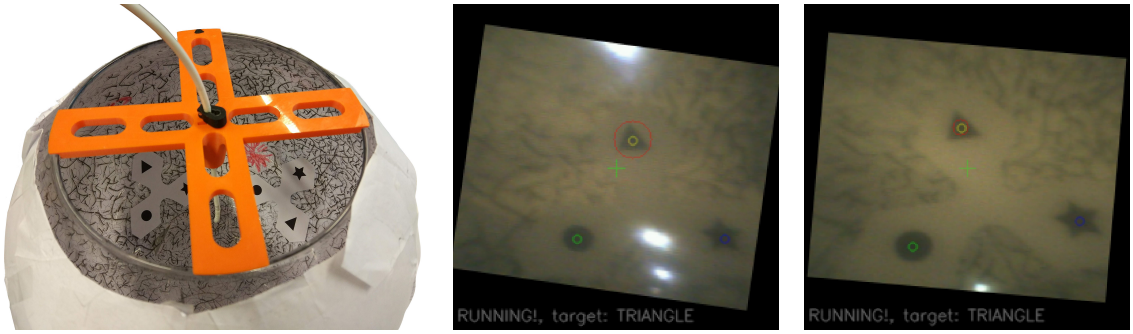


Figure 5.3: The spherical phantom used in the pointing task experiment (left image) and visual feedback seen by the experiment participants (center image: experiment using ID = 2.0, right image: using ID = 3.0).

The order in which targets need to be reached is kept the same: first to circle, then to triangle, then to star, and then back to circle, and so on. The distance between targets is set at 8 cm, and the target tracks are placed 16.5 cm away from the endoscope camera on the inside of the spherical phantom, at around 50° elevation angle from the vertical axis. After 20 targets are reached, one session is concluded.

There are 2 tracks used in this experiment, to remove task muscle-memory effect. The overall shape of the tracks are both the same (triangle), however the direction is changed to counter-clockwise using the second track. The same pointing task experiment will be done for both usage types: head-scope system and manual usage.

Target center is detected using a binary large object (blob) detector, from which then the center is calculated. The size of the target diameter is determined by the starting distance to target and ID, and the relation is given by Equation 2.6.

A training session is done for each system to remove experiment participant's learning curve. A trial was done beforehand with 2 subjects to investigate the amount of training time needed for the measured motion time to level-off. From these 2 trials, the learning curve for using the head-scope system cannot be perceived as the motion time measurement is almost flat from the start.

Nevertheless, 2 training sessions are performed for the head scope usage with ID of 2 and 3, while for the manual usage experiment, one training session is done with ID of 2 for each participant. The full schedule of the experiment for each participant is presented in Table 5.1. The amount of manual usage trial is less than head-scope trial due to the limited clinician's avail-

[1] Technical Medicine is a Masters level programme in which students learn to integrate technologies within the medical sciences to improve patient care.

System	Trial type	Index of Difficulty (ID)	N	Track
Head-scope	training	2.0	1 × 20	1
Head-scope	training	3.0	1 × 20	2
Head-scope	test	2.0	1 × 20	1
Head-scope	test	3.0	1 × 20	2
Manual usage	training	2.0	1 × 20	1
Manual usage	test	2.0	1 × 20	2
Manual usage	test	3.0	1 × 20	1
Hololens-scope	training	2.0	1 × 20	1
Hololens-scope	test	2.0	1 × 20	2
Hololens-scope	test	3.0	1 × 20	1

Table 5.1: Experimental protocol for training and measurement. Note that the last 3 sessions with the Hololens-scope were conducted only with non-clinician participants due to the time availability of the surgeons, latency limitation, and Hololens application development progress at the time.

able time during the visit to UMCG. In total 4 clinicians participated in this experiment, and 3 of them are active surgeons in UMCG.

For this experiment, a questionnaire is also asked (listed in Appendix C).

Figure 5.4: Setup during the use of head-scope system (left) and manual system (right) in the pointing task experiment with the clinicians in UMCG.

Manual flexible scope usage

For manual control the handle of the endoscope can be held in either hand. With the thumb, the control lever can be moved. The control lever is used to flex and extend the tip of the endoscope in the vertical plane. Moving the control lever downward will make the tip bend anteriorly (flexion). Moving it upward will make the tip bend posteriorly (extension). The endoscope camera can be rotated by rotating the handle.

Head-controlled endoscope usage

For the head-controlled endoscope, a headband is placed on the participant's head, specifically with the IMU sensor on the left temple of the user. The movement of the image on the screen will correspond with the head movement: when the head moves up and down, the image moves up and down, and the same for left and right. The camera motion is only activated when the footswitch is pressed.

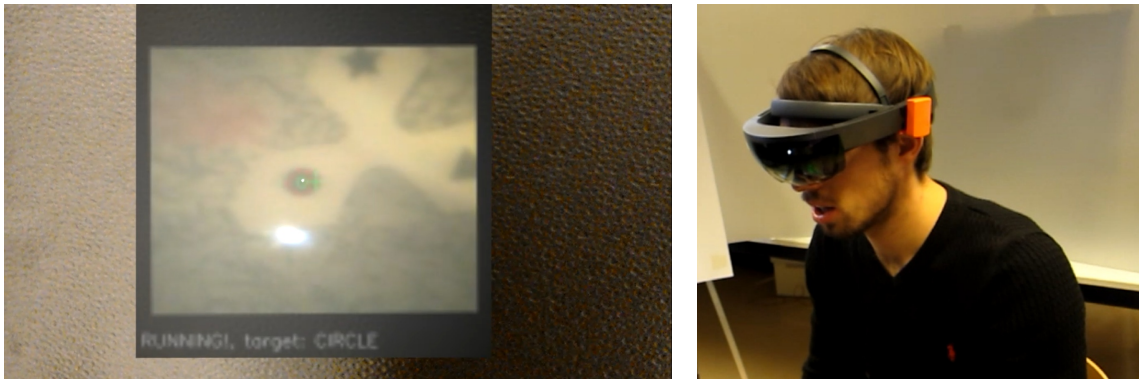


Figure 5.5: View from the headset (left) and a user wearing the Hololens head-controlled system (right) during the pointing task experiment.

Head-controlled endoscope usage with Hololens as visual output

For head-scope with Hololens, the Hololens is adjusted to the participant's head, and the IMU is placed on headset's left side, aligned with the display glasses plane orientation. The image output is presented to the user on a square image plane placed 30 cm away, covering most of the height of Hololens' display glasses. This image plane will stay directly on the user's sight, adjusting its position based on headset orientation. Method of operation for this system is identical to the normal head-controlled endoscope (with headband).

6 Results

In this chapter, results on the technical validation (Section 5.1) and pointing task experiments (Section 5.2) are presented.

6.1 Technical validation of system modules

Technical validation of the head-controlled endoscope system consists of tip deflection angle calibration, estimated rotation angle validation of the image rotation compensation module, and image output latency measurements.

6.1.1 Tip movement calibration

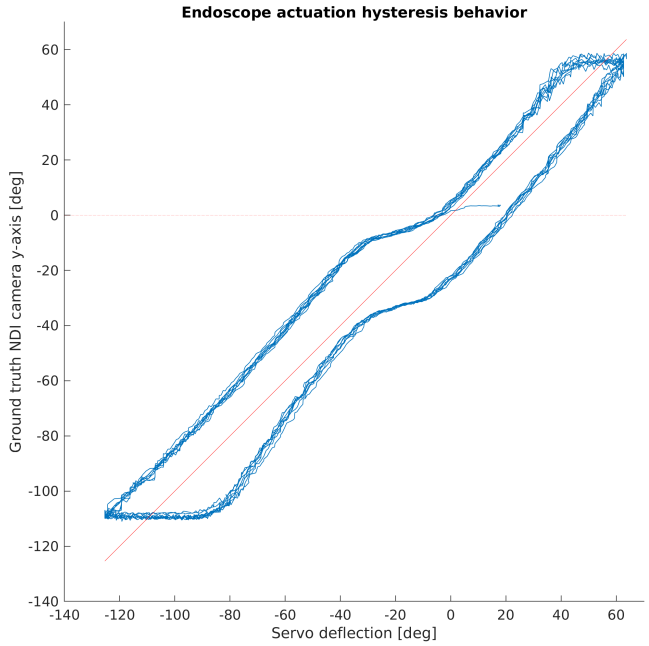


Figure 6.1: Deflection input given to the servo versus the measured deflection using tracker attached at the endoscope tip. Servo angle to deflection angle map in the implementation is given as linear relation shown by the red line.

The hysteresis behavior is apparent in the deflection axis, due to backlash in the actuation mechanism and the thumb control lever of the endoscope. Some non-linearities are also present around $-20°$ deflection due to the spring-like mechanism of the thumb lever which is not zeroed at straight tip deflection. Since a servomotors is used, only angle input is prescribed and force applied onto the thumb lever cannot be controlled, hence the error at this region.

There is a small deflection in the other (non-controllable) deflection axis ($\theta_{\text{peak-to-peak}} \leq 9.2°$) due to the imperfect / non-centered cable mechanism inside of the flexible endoscope.

6.1.2 Image rotation compensation accuracy and runtime performance

The estimated rotation angle of both compensation methods, compared against a ground-truth angle, is presented in Figure 6.2. The optical flow based rotation compensation has a big error due to drift, however the details (higher frequency component) in the motion look very similar to the ground-truth.

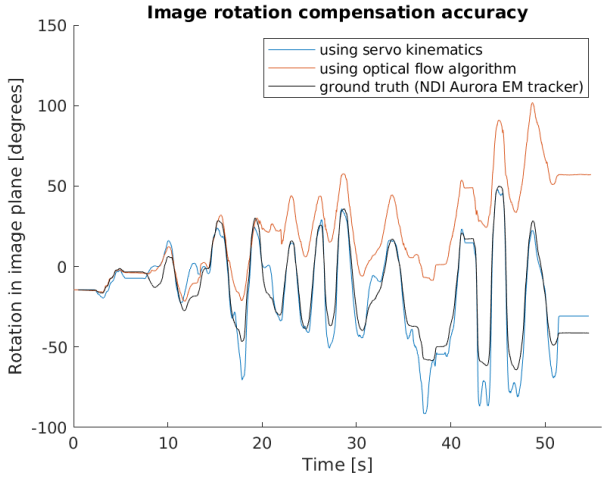


Figure 6.2: Image rotation angle for the image rotation compensation module obtained in 3 ways: using optical flow algorithm, using kinematics from servo, and ground truth using NDI electromagnetic tracker.

While compensation using servo position information does not suffer from drift, it is not accurate in the finer motion details because of the hysteresis behavior in the deflection axis, and also because of the uncertainties from the cable compliance.

Image compensation method	Run-time per loop
Using kinematics / servo position	5.93 ± 1.61 ms
Using SimpleFlow optical flow algorithm	30.47 ± 17.71 ms

Table 6.1: Processing time needed by each compensation algorithm implementation for one frame / message iteration. This processing time includes the time the software needs to publish the image into a ROS message.

The image rotation compensation run-time presented in Table 6.2. The rotation compensation ROS node is running on a laptop with an *i7 6700HQ* 4-cores 8-threads CPU.

6.1.3 Image pipeline latency

Output	Photon-to-photon latency
Ambu monitor	43.7 ± 11.9 ms
Raw input image in ROS	254.8 ± 28.1 ms
Rectified image (using kinematics from servo)	265.6 ± 46.5 ms
Hololens output	358.3 ± 117.9 ms

Table 6.2: Photon-to-photon latency, which is the time between light entering the camera and image output at respective devices, is measured at every image output location in the system.

Photon-to-photon latency at every image output is presented in Table 6.2. The video capture device introduces most of the latency observed in the system, because the analog image exported by the Ambu monitor has to be converted and encoded into digital format for further processing.

Wireless communication and internal processing of the Hololens also add significant amount of delay. Detectable jitter and skipped frames are also observed in the Hololens' image output.

6.2 Pointing task experiment

The pointing task experiment with clinicians does not include the use of Hololens headset due to the latency limitation, time availability, and Hololens application development progress at the time. Furthermore, image rotation compensation using kinematics from servo was employed for all pointing task experiments.

Clinician reaching time results

The pointing task experiment detailed in Section 5.2 is done with 4 clinicians from the cardiothoracic department at the UMCG. The distributions of reaching times for each clinician are presented in Figure 6.3.

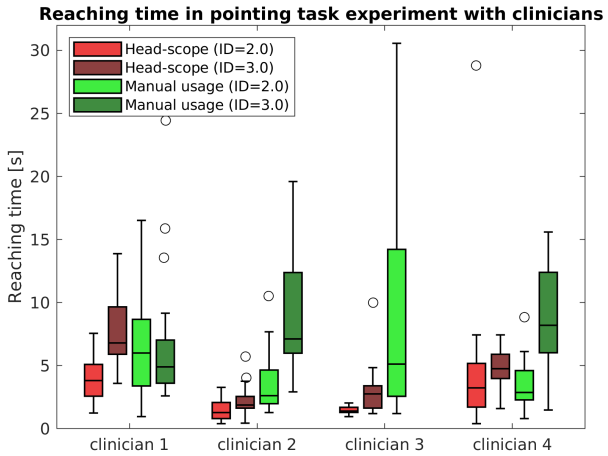


Figure 6.3: Clinicians reaching time measurement in using the head-scope system with standard monitor display output (red) and manual scope (green). The small circles indicate measurement points that are considered as outliers. Manual usage with higher difficulty level (ID= 3) for clinician 3 was not conducted due to his availability at the time.

Statistical analysis using two-sample t-test is conducted to look for difference in reaching time mean between head-controlled scope and manual usage. The statistical tests were conducted two times for an ID of 2 and an ID of 3. Significance level $\alpha = 0.05$ is used for all statistical tests.

Results indicate a significantly faster reaching time within clinician participants ($N = 3$) at higher difficulty ID= 3 using head-controlled scope ($\mu = 4.44\,\text{s}$, $\sigma = 2.39\,\text{s}$) over using manual scope ($\mu = 8.23\,\text{s}$, $\sigma = 1.01\,\text{s}$), $p = 0.0435$. The difference in mean at lower difficulty level (ID= 2) within clinician participants is not significant.

Some correlation between reaching time and questionnaire parameters such as age and gaming experience was observed. This observation will be discussed in Section 7.1.

Trained non-clinician reaching time results

Pointing task experiments identical to the previous section were carried out with 8 non-clinicians who were mainly Technical Medicine students who had completed a training in clinical insertions and endoscopy. For this set of participants, the head-controlled Hololens endoscope is added as a third endoscopic system to be tested. Reaching time results are presented in Figure 6.4.

For non-clinician participants ($N = 8$) at lower difficulty level (ID= 2), the two-sample t-test indicates significantly faster reaching times using the head-controlled scope with display monitor output ($\mu = 1.40\,\text{s}$, $\sigma = 0.416\,\text{s}$) over using manual scope ($\mu = 1.90\,\text{s}$, $\sigma = 0.413\,\text{s}$), $p = 0.0290$. Reaching time is also significantly lower at higher difficulty level (ID= 3) using the head-

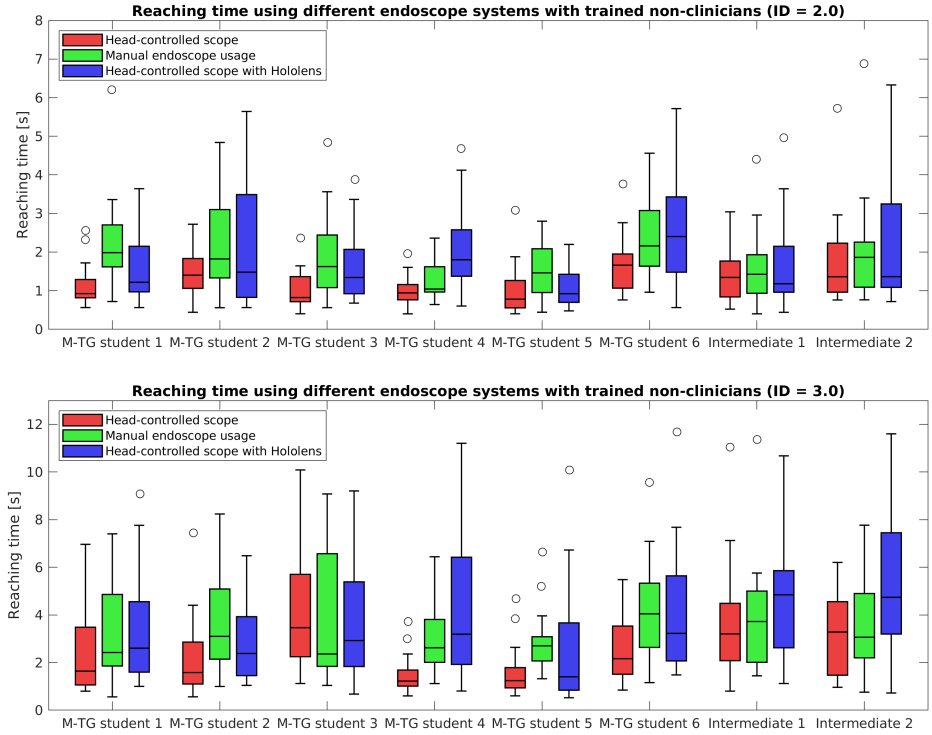


Figure 6.4: Motion time results using head-scope system (red), manual usage (green), and head-scope system with Hololens as visual output (blue). Reaching time for the task with ID of 2 is shown on top image, while ID of 3 is shown on bottom image.

controlled scope ($\mu = 2.71\,\text{s}$, $\sigma = 0.997\,\text{s}$) over using manual scope ($\mu = 3.67\,\text{s}$, $\sigma = 0.540\,\text{s}$), $p = 0.0351$. Differences in reaching time with the Hololens head-controlled scope system against manual usage are not statistically significant for both difficulty levels.

Reaching time difference between the head-scope (monitor output) and manual usage with student participants are bigger compared to the more trained endoscopic course lecturers. This may signify a lower learning time required for the head-scope system.

Questionnaire results

Questionnaires (Appendix C) were given to the clinicians and non-clinician participants to assess the user experience of the manual and head-controlled (with monitor) endoscope system. The questionnaire responses from all participants ($N = 12$) are presented in a Likert chart shown by Figure 6.5.

Most notable differences in user responses between the two systems are regarding the confidence felt while using the system, amount of training needed, and user-friendliness of the respective endoscope systems. User responses indicate that participants felt more confident in using the head-controlled system compared to the manual flexible endoscope. Participants also think that less training is needed with the head-scope and most people would quickly learn how to use the head-scope system. The head-controlled system is easier to use and more intuitive according to the user responses compared to the manual usage.

However, regarding the amount of fatigue experienced by the user, responses indicate that both systems were not fatiguing. Most participants did not feel exhausted by the end of the experiment sessions, even though the experiment took 40 minutes in total per participant. Most participants also think that little instruction is needed to be able to properly use both the manual and head-scope endoscope systems.

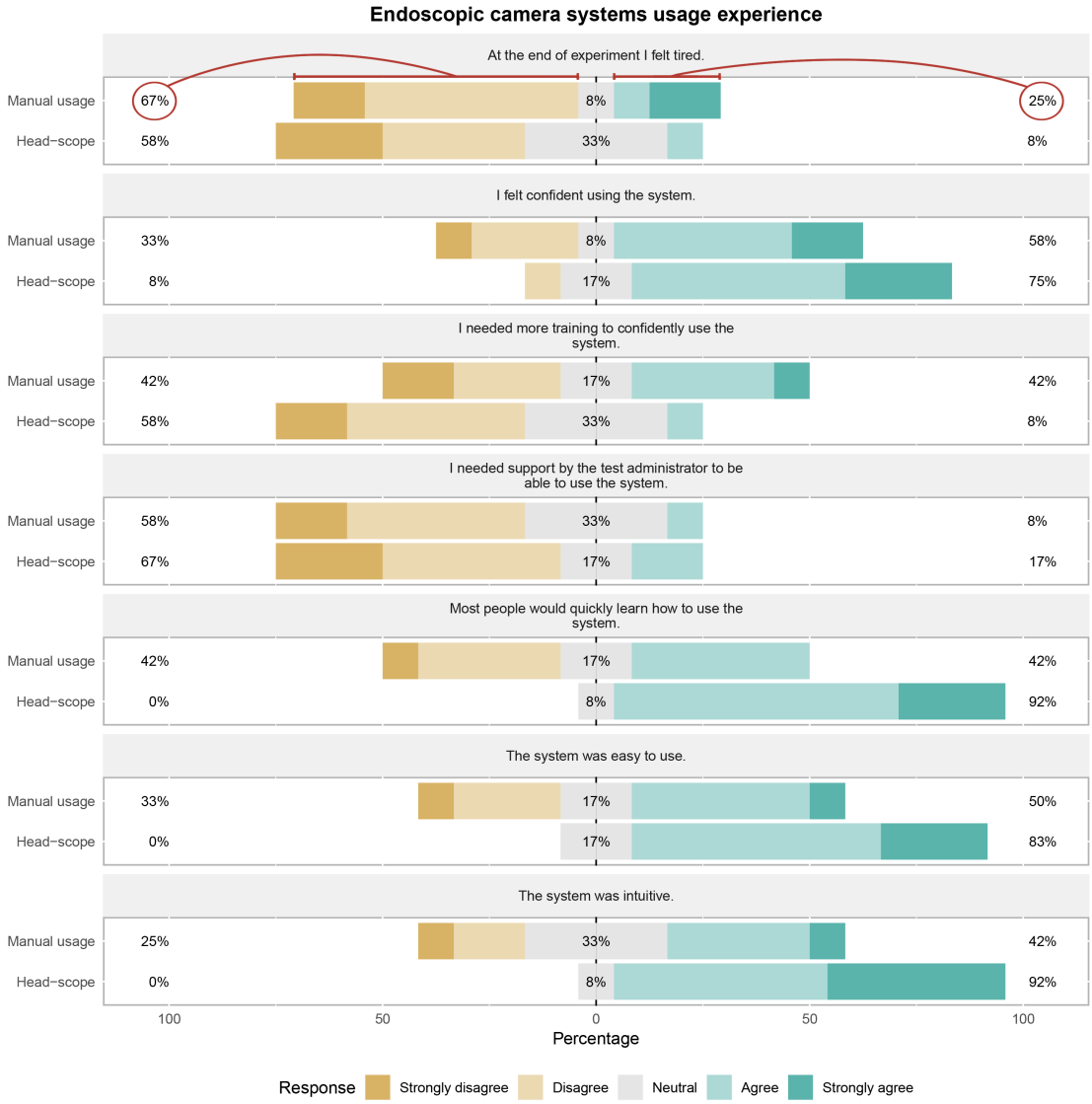


Figure 6.5: Response for the questionnaires on user experience when using the manual flexible endoscope system (Ambu aScope 3 bronchoscope) versus the head-scope system. The percentage shown to the left, center, and right of the chart indicate the sizes of the 'disagreeing zones', 'neutral', and 'agreeing zones' respectively.

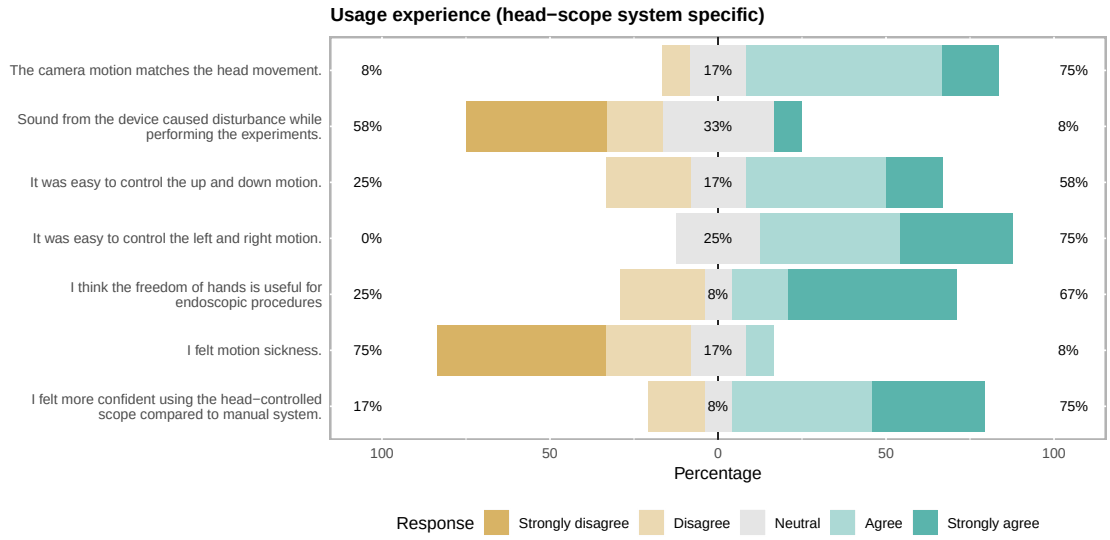


Figure 6.6: Responses for the questionnaires on user experience specific for the head-scope system.

Some head-controlled system specific questions were asked in the questionnaire (shown in Figure 6.6), such as the perceived motion, sound, motion sickness, and whether they think that hand freedom during surgery is useful. Responses indicate that the perceived image motion matches their head movement, while only one participant did not sense matching image movements. Only 1 out of 12 participants thinks that the sound from the head-scope system disturbed their concentration during the experiment. Participants think that controlling the left and right is easier compared to up and down motion. This agrees with the result shown in Section 6.1.1, as up and down motion mostly correspond to the endoscope deflection axis which suffers from actuation mechanism backlash. Out the of 4 clinicians participated in this study, 2 strongly agree and one agree that hand freedom is useful in endoscopic procedures, while one other clinician disagrees with this statement. One out of 12 participants felt motion sickness while using the head-controlled endoscope. Overall, participants felt more confident in using the head-controlled endoscope compared to the manual system.

7 Concluding Remarks

Discussion points over the whole project and the conclusion are presented in this chapter. Afterwards, recommendations for future work are listed in the subsequent section.

7.1 Discussion

Some discussion points in the scope of the whole project are presented:

- Correlation between age and reaching time performance using the head-scope system was observed for both clinician and non-clinician participants. Older participants tend to have slower reaching time performance. This age and reaching time correlation is not apparent with the flexible endoscope manual use.

  Correlation between gaming expertise and reaching time was also apparent in all endoscope systems tested in the experiments. Technical medicine / M-TG student participants 1, 2, 3, and 5 (from Figure 6.4) have indicated spending more than three hours per week on gaming. All, except M-TG student 3, scored faster than average reaching time (averaged over all 3 systems).

- Forty minutes overall experiment time might not be enough to induce fatigue when using both endoscope system (head-scope and manual). Further research is required to compare the amount of fatigue caused by each endoscope system in a task with long duration (to simulate long operating hours).

- The head-scope system developed in this project has more than 0.25 seconds *photon-to-photon* latency, limiting the human performance while using the system. However, based from the remarks of the clinicians who participated in the experiment, even though the head-scope system suffers from latency and hysteresis problems, the system is not more difficult to use compared to the rigid endoscope they are using in the OR.

- A suggestion has been proposed regarding the calibration method: using a spherical target (a checkerboard pattern laid onto the inside of a spherical dome), instead of a flat checkerboard pattern to improve optical flow accuracy. This will improve the clustering of the flow vector, therefore helps in improving the accuracy of the rigid-body transform estimation.

  However for practical usage, the user would expect an image similar to what human eyes perceive as visual feedback from the head-scope system. Calibrating the camera using a spherical target would violate this expectation, since a line on a curved surface (seen exactly from the center of the sphere) will be displayed as straight. Also from practical point-of-view, a spherical phantom is not an exact representation of a human body cavity. Therefore, a standard calibration using pinhole camera model with flat checkerboard target is used for our system.

7.1.1 Limitations

Some challenges and hardware limitations was encountered during the development of the head-controlled endoscopic camera system:

- The image output suffers from significant amount of delay (more than 0.25 s due to non-ideal image pipeline. The raw image is exported through the proprietary Ambu monitor, and captured with an (analog) video capture device. A better endoscope camera with low latency image pipeline is needed to solve this problem.

- The endoscope image quality, specifically the resolution (220×180 px) and color contrast, is very limiting for the optical flow algorithm and blob detection process. The currently used flexible endoscope is only meant for single use, hence the poor quality of the output image.

  We tried attaching a better quality (720p) endoscopic camera to the Ambu flexible endoscope, but the tip deflection mechanism is very flimsy and fragile. The stiffness of the endoscope tip mechanism cannot support the weight of the better quality camera.

- The rotation axis actuation mechanism cannot rotate continuously, therefore the reachable orientation space of the endoscope tip is not full 360° degrees. Therefore, the currently developed head-scope system is only suitable for a specific use-case (diamond port positioning where camera does not need to rotate 360°).

7.2 Conclusion

A head-controlled endoscopic camera system with a footswitch control has been successfully realized in this project. Several validation tests and pointing task experiments were conducted at two different index of difficulty levels with clinicians and non-clinician participants. Questionnaires were used to evaluate the user experience while using the endoscope system.

Head position control with the help of an image rotation compensation to stabilize the output image orientation using servo kinematics was the chosen implementation to make the system more intuitive and easy to use. The intuitiveness, easy-to-learn, and user-friendliness of the head-controlled endoscope system were verified by the questionnaire responses from the participants. The head-scope system suffers from high *photon-to-photon* latency (265.6 ± 46.5 ms using a monitor and 358.3 ± 117.9 ms with the Hololens), mostly due to hardware limitation in the image pipeline.

To what extend, can a head-controlled endoscopic camera system offer a solution to the limiting factor in MIS?

The head-scope system developed during this project is a proof of concept that a head motion controlled endoscope offers solution to the limitation in MIS procedure. The results from the pointing task experiments show that the developed head-controlled endoscope has significantly faster reaching time performance in the high index of difficulty task with clinician participants ($p = 0.0435$), and in both ID tasks with non-clinician participants ($p = 0.0290$ for ID= 2 and $p = 0.0351$ for ID= 3), against the manual usage of the flexible endoscope. Moreover, the questionnaire responses indicate that the head-scope system is acceptable among clinician and non-clinician participants.

More importantly, several limitations would be resolved by the head-scope system, namely, more direct camera control by the surgeon, better ergonomics in the OR due to fewer personnels, and reduced post-operative pain induced by the pressure from the rigid endoscope to the rib cage. Indirect benefits, such as better vision around corners and maneuverability inside of the patient's body can be realized with the head-controlled endoscope, in contrast to the rigid nature of the currently used rigid endoscope.

7.3 Recommendations for future work

Further work should be conducted to improve the performance in various parts of the head-controlled endoscope system. Recommendations for future work on the head-scope system are listed in this section:

- Better image rotation compensation performance can be achieved using sensor fusion approach, combining the angular velocity from optical flow and the image rotation us-

ing kinematics. Significant accuracy increase is very likely, as the validation result in Section 6.1.2 indicates that the estimate using optical flow is accurate in the finer motion details (but suffers from drift).

Even further, the translation part of the estimated rigid-body transform can be used for hysteresis compensation on the actuation of the endoscope tip, similar to the approach presented by Reilink (2013).

- Hysteresis compensation on the actuation of the endoscope's deflection axis can be done simply by modelling the hysteresis behavior presented in Section 6.1.1. Using this approach, accuracy in the perceived up and down motion can be improved.

- Hololens built-in positioning system can be used as head-orientation input, instead of using the IMU. Negligible latency increase should be expected compared to the current implementation, as the currently used IMU is also wireless. In terms of accuracy, further research needs to be conducted to measure the error in Hololens orientation estimate compared to the Xsens IMU's orientation output.

- Other dense optical flow algorithms such as Brox (Brox et al., 2004) or Farneback (Farnebäck, 2003) are also suitable for our implementation. These algorithms might have advantage against our current implementation of SimpleFlow algorithm, since OpenCV's optical flow library has the GPU implementation API already available for both.

  Brox et al. (2004) in their paper, compared their own algorithm against Farneback and has better accuracy result in a standardized optical flow benchmark (Lynn Quam's Yosemite sequence (Heeger, 1987)). However, accuracy comparison against SimpleFlow algorithm in a standardized benchmark cannot be found in literature. Though, moving from central processing unit (CPU) to GPU implementation would already give significant boost in computation speed, and performance increase of the image rotation compensation module is very likely.

  More recent optical flow algorithms are also available outside of OpenCV library. There is a published ranking for optical flow algorithms (Geiger et al., 2015) using KITTI Vision Benchmark Suite (KITTI) flow dataset (Menze and Geiger, 2015). The ranking is ordered based on the flow vector accuracy, and many open-source optical flow algorithms are listed here. At the time of writing, the top ranked mono-camera optical flow algorithms that are suited for real-time computation are all GPU-based and use machine-learning algorithms either to directly estimate the flow, or to fuse consecutive flow frames to get a better estimate. Brox, Farneback, and Pyramid-LK algorithm implemented in OpenCV are ranked 94^{th}, 96^{th}, and 99^{th} respectively in this list.

- Regarding the potential clinical application of the system, 3^{rd} degree of freedom is needed to control the insertion depth of the endoscope camera. This can be achieved simply by adding a linear stage to the whole actuation mechanism.

A Appendix 1: Endoscope Camera Calibration

Camera calibration is an essential step in order to determine the distortion and camera intrinsic parameters which are used in the 'un-distortion' step of image-preprocessing. This step is important as the endoscope camera suffers from lens distortion, specifically barrel distortion. Without the un-distortion step, the estimated flow vectors may not point to the same direction in a pure translation motion, since straight lines in real world may appear curved in the image.

To 'un-distort' an image, 2 things are needed: the camera matrix (specifically the intrinsic parameters) and the distortion coefficients.

A.1 Calibration models

A.1.1 Pinhole camera model and the intrinsic parameters

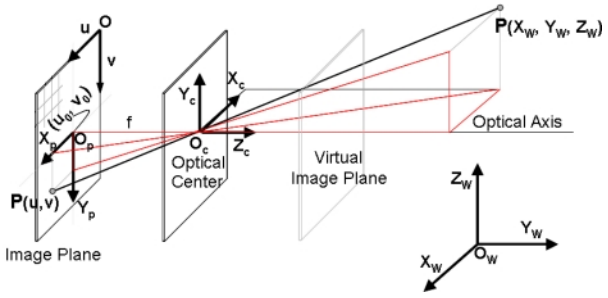


Figure A.1: Image projection using the pinhole camera model

In essence, a camera matrix projects 3D world coordinates $[x_w \ y_w \ z_w]^\intercal$ into 2D coordinates $[u \ v]^\intercal$ in the image plane. The camera matrix $\mathbf{P}$ is a 3×4 matrix, which is the product of the intrinsic matrix and the extrinsic matrix.

$$\alpha \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{P} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \quad \text{with} \quad \mathbf{P} = \begin{bmatrix} f_u & s & p_u \\ 0 & f_v & p_v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} & R_{13} & t_1 \\ R_{21} & R_{22} & R_{23} & t_2 \\ R_{31} & R_{32} & R_{33} & t_3 \end{bmatrix}, \tag{A.1}$$

where α is a scale factor for the image point.

The intrinsic matrix has a skew parameter s, which can be set to 0 since modern camera sensor has negligible skew (Zhang, 2000). The rest consists of focal lengths f_u and f_v, measured in pixel, and the location of principal point (the pinhole location projected into the image plane).

The extrinsic matrix consist of rotation matrix $\mathbf{R}$ and translation vector $\mathbf{t}$, which defines the pose of the camera with regard to the world frame.

A.1.2 Distortion models

The projected image result in the image plane most likely will have optical distortion, where straight lines in a scene are no longer straight in the image. This is due to the camera model assumes the incoming light always converges at a point, meanwhile real world camera lenses are not infinitesimally small. Therefore, there will be a difference in magnification depending on the distance to the optical axis (radial distortion). Some skew or positioning imperfections may also cause the image to have a tangential distortion.

Radial-tangential distortion model

Radial-tangential distortion model, also called radtan or plumb bob model, is a polynomial camera distortion model with 3 radial and 2 tangential parameters (Heikkila and Silven, 1997).

$$\begin{aligned} u &= u_d(1 + k_1 r_d^2 + k_2 r_d^4 + k_3 r_d^6) + 2p_1 u_d v_d + p_2(r_d^2 + 2u_d^2) \\ v &= v_d(1 + k_1 r_d^2 + k_2 r_d^4 + k_3 r_d^6) + 2p_2 u_d v_d + p_1(r_d^2 + 2v_d^2), \end{aligned} \quad \text{where} \quad r_d = \sqrt{u_d^2 + v_d^2}. \tag{A.2}$$

Some calibration tools output 4 radtan distortion coefficients (2 for radial and 2 for tangential), and so $k_3 = 0$.

Equidistant distortion model

The equidistant distortion model (Kannala and Brandt, 2006), is a 4 parameters distortion model (as implemented in Kalibr), that models the radial distortion using a polynomial function of the angle between incoming ray and the principal axis.

$$\theta_d = \theta(1 + k_1\theta^2 + k_2\theta^4 + k_3\theta^6 + k_4\theta^8), \tag{A.3}$$

$$\theta = \arctan(r), \tag{A.4}$$

$$r = \frac{\sqrt{x_c^2 + y_c^2}}{z_c}, \tag{A.5}$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & s & p_u \\ 0 & f_v & p_v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{\theta_d}{r}\frac{x_c}{z_c} \\ \frac{\theta_d}{r}\frac{y_c}{z_c} \\ 1 \end{bmatrix}, \tag{A.6}$$

where x_c, y_c, and z_c are coordinates in camera frame (after the multiplication of extrinsic matrix).

This equidistant distortion model is used for our live-setup implementation using the fisheye lens.

ATAN distortion model

ATAN distortion model, or also called fov distortion model is a distortion model with 1 parameter (field of view ω), modelling the radial distortion of the lens. This distortion model is not adequate for modelling the complex distortion from fisheye optics (Devernay and Faugeras, 2001).

$$r_u = \frac{\tan(r_d\omega)}{2\tan\frac{\omega}{2}} \tag{A.7}$$

A.2 Calibration tools

A.2.1 MATLAB camera calibration toolbox

MATLAB provides a streamlined camera calibrator app within the computer vision system toolbox.

+ Easy-to-use camera calibration procedure, and if the camera driver is supported by MATLAB, calibration image can be captured directly using the app.
+ Supports single and stereo camera calibration.
+ Very descriptive GUI visualizing the calibration target pose and mean error for each calibration image.
+ Supports radtan and equidistant (fisheye) distortion model.
– MATLAB with computer vision system toolbox needs to be installed, which takes quite a lot of disk space.

A.2.2 `camera_calibration` package in ROS

The `camera_calibration` package is a part of `image_pipeline` stack from ROS, allowing easy calibration of monocular or stereo camera using a checkerboard calibration target. Some advantages and disadvantages of `camera_calibration` compared to other camera calibration tools are:

+ Nice GUI which tells calibration certainty in x and y axes.
+ Automatic commit of calibration parameters, writing it into a YAML file which can be loaded directly with ROS camera driver using the `camera_info_url` parameter.
– Supports only the radtan distortion model, does not support equidistant model (for fisheye cameras).

A.2.3 Kalibr

Kalibr is a multi-functional camera calibration tool, integrated into ROS, used for not only estimating the intrinsic and distortion parameters, but also for estimating extrinsic transformations if multiple camera is used.

+ Powerful and multi-functional calibration tool which support pinhole and omni camera model, with radtan, fov, or equidistant distortion model.
+ Automatically creates a calibration report which details the reprojection errors and residuals.
+ Has support for multiple camera setup.
+ Has many complementary tool such as focus calibration, and calibration validation.
– Does not automatically commit to ROS camera driver, only produces a `.yaml` file which does not have the same format as ROS `sensor_msgs/CameraInfo` (which `image_undistort` package comes handy).

A.3 Endoscope camera calibration using Kalibr

In this section, practical steps for endoscope camera calibration are presented.

A.3.1 Camera and calibration target setup

The Ambu aScope 3 Broncho Regular has 85° field of view and operating depth of field range between 6-50 mm. Camera settings (shutter speed, aperture, and sensor speed) are not configurable. The recorded image frame-rate can be configured from OpenCV side by changing the time interval before reading the next frame from capture device memory. The output analog image from the Ambu monitor is sent at 25 fps (PAL encoding). The calibration image stream is recorded into a *rosbag* at 4 fps.

The calibration target is set at distance 8-20 cm, similar to the distance to the pointing target in the experimental setup (Section 5.2). The calibration target is a 8-by-7 checkerboard pattern with 10 mm square size, attached into a rigid flat surface.

A.3.2 Calibration procedure

To estimate the intrinsic and distortion parameter of the endoscope camera, a static calibration needs to be performed. The endoscope needs to be in static position, while the checkerboard target needs to be moved covering the whole field of view (FoV) of the camera. The output of this static calibration is the estimated intrinsic camera parameters & distortion parameters.

To do static calibration, first install Kalibr by following the installation steps in https://github.com/ethz-asl/kalibr/wiki/installation. Record the image stream into a *rosbag* file at 4 Hz, and make sure that the calibration target is moved slowly at the required distance to avoid motion blur. Set up the calibration target parameters in the `checkerboard_7x6_10mm.yaml` parameter file:

```
target_type: 'checkerboard' #gridtype
targetCols: 7               #number of internal chessboard corners
targetRows: 6               #number of internal chessboard corners
rowSpacingMeters: 0.01      #size of one chessboard square [m]
colSpacingMeters: 0.01      #size of one chessboard square [m]
```

Finally, run the calibration using:

```
kalibr_calibrate_cameras --checkerboard\_7x6\_10mm.yaml --bag [
    filename.bag] --models pinhole-radtan --topics /image/cropped
```

The static calibration is done using pinhole-radtan model. The calibration is performed 6 times and result with the lowest reprojection error (< 0.2px) is used. Even though the resulting reprojection error is small, the calibration results are not consistent. This inconsistency might be caused by small usable image resolution (220×180 px) and focus distance that is very close to the endoscope camera (<5 cm).

B Appendix 2: Code Repository

The source codes developed during this project can be accessed (internal) in the following RAM GitLab repository:

- `optical_flow_head_scope`:
  (https://git.ram.ewi.utwente.nl/makyx/optical_flow)
- `herkulex_servo_controller`:
  (https://git.ram.ewi.utwente.nl/makyx/herkulex_servo_controller)
- `head_scope_middleware`:
  (https://git.ram.ewi.utwente.nl/makyx/head_scope_middleware)
- `ros_footswitch`:
  (https://git.ram.ewi.utwente.nl/makyx/ros_footswitch)
- `awinda_monitor`:
  (https://git.ram.ewi.utwente.nl/makyx/awinda_monitor)
- `head-scope-unity`:
  (https://git.ram.ewi.utwente.nl/makyx/head-scope-unity)

An R script to create the questionnaire response plot (Likert chart) automatically based on Google form questionnaire results (Goggle sheets) can be accessed in

- `questionnaire-likert-chart`:
  (https://git.ram.ewi.utwente.nl/makyx/questionnaire-likert-chart)

Documentation on how to install and use each ROS package can be found in the respective repository.

C Appendix 3: Head-scope pointing task experiment documents

C.1 Instruction manual for participants

C.1.1 Introduction

First of all thank you that you want to participate in this research. In order to prepare for the tests this instruction has been made. If you have any questions after reading this please ask them.

The goal of this project is to evaluate a new method for controlling an endoscope by using head movements. During this research we compare manually controlling an endoscope versus head movements controlling an endoscope.

C.1.2 What do you have to do?

During the tests you see an image with 3 kinds of target figures on a screen: circle, triangle, and star target (Figure 1). In each session it is the goal to move the centre crosshair to the inside of the projected red circled target on the figures and touch it for at least 0,4 second. If this is achieved the next appears which has to be touched. The first target will be on the circle figure, then triangle, star, and then back to circle figure.

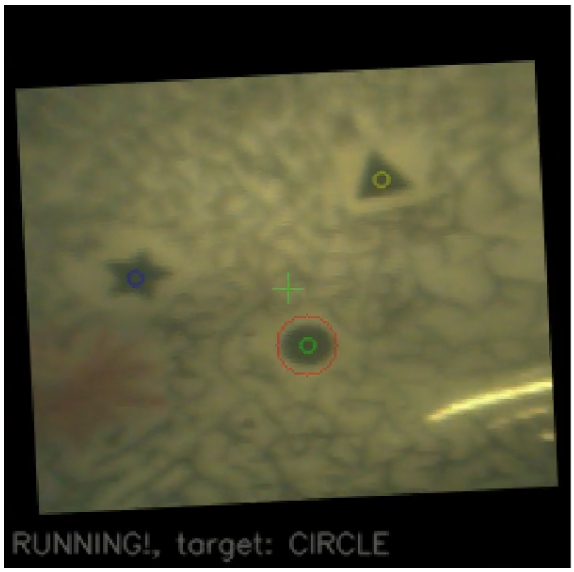


Figure C.1: Screenshot of the pointing task experiment visual output.

Before the test starts you get the chance to practice with the different controlling options in order to avoid a learning effect in the measurements.

During the whole experiment there is the opportunity to ask questions. After the test follows a questionnaire. The tests will take about half an hour.

C.1.3 Manually controlling an endoscope

The handle of the endoscope can be held in either hand. Use the thumb to move the control lever. The control lever is used to flex and extend the tip of the endoscope in the vertical plan. Moving the control lever downward will make the tip bend anteriorly (flexion). Moving it upward will make the tip bend posteriorly (extension). By rotating the handle you can rotate the endoscope.

C.1.4 Head movements controlling an endoscope

By using your head it is possible to move the blue centre point over the image, as moving a virtual camera over the surface. By looking up and down the camera moves up and down, and by moving your head from left to right the camera also moves from left to right. Then there is also a clutch in the form of a foot pedal present with which you can switch the control on and off. When the footswitch clutch is pressed, the camera motion is engaged and otherwise is not.

C.1.5 Things to pay attention for

- Try to perform the tests as fast and as accurate as possible.
- Try to think loudly during the tests, this can help to clarify your feeling about a specific aspect.
- If you want to stop with the tests at any time please tell this.
- Finally try not get distracted.

C.1.6 Finally

The whole procedure is captured by a camera, in order to review the experiments afterwards. Please indicate in advance if you don't want this.

C.2 Consent and questionnaire form

Questionnaire Experimental Setup

* Required

Informed consent

1. **I have read and understood the Instructions for experiment 'Camera control by head movements'. I had the opportunity to ask questions about it and any questions that I have asked have been answered to my satisfaction. I consent voluntarily to participate in this research. ***

   Check all that apply.

   ☐ Agree

Personal data

2. **Name:**

   _____

3. **Occupation:**

   _____

4. **Age:**

   _____

5. **Gender:**

   Mark only one oval.

   ◯ Male
   ◯ Female

6. **Do you have any eye disorders?**

   Mark only one oval.

   ◯ Yes
   ◯ No

7. **Do you have any neuromuscular dysfunction?**

   Mark only one oval.

   ◯ Yes
   ◯ No

8. **Do you have any computer gaming experience?**
Mark only one oval.

- ◯ No
- ◯ Yes, < 1 hour per week
- ◯ Yes, 1 - 3 hours per week
- ◯ Yes, > 3 hours per week

9. **Do you have any endoscopic procedure experience?**
Mark only one oval.

- ◯ Yes *Skip to question 10.*
- ◯ No *Skip to question 13.*

Endoscopic procedure experience

10. **What type of endoscope do you use in your experience?**
Check all that apply.

- ☐ Flexible scope
- ☐ Rigid scope
- ☐ Other: _____

11. **What is your role in endoscopic procedures?**
Mark only one oval.

- ◯ Surgeon
- ◯ Assisting

12. **What is the estimated number of endoscopic procedure you have done?**
Mark only one oval.

- ◯ < 30
- ◯ 30 - 100
- ◯ > 100

Experiment questionnaire

13. **Head-controlled scope movements:**
Mark only one oval per row.

	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
I felt motion sickness.	◯	◯	◯	◯	◯
The camera motion matches the head movement.	◯	◯	◯	◯	◯
It was easy to control the left and right motion.	◯	◯	◯	◯	◯
It was easy to control the up and down motion.	◯	◯	◯	◯	◯

14. **Head-controlled scope experience:**
Mark only one oval per row.

	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
The system was intuitive.					
The system was easy to use.					
I needed support by the test administrator to be able to use the system.					
Most people would quickly learn how to use the system.					
I felt confident using the system.					
I needed more training to confidently use the system.					
At the end of experiment I felt tired.					
Sound from the device caused disturbance while performing the experiments.					
I think the freedom of hands is useful for endoscopic procedures					
I felt more confident using the head-controlled scope compared to manual system.					

15. **Manually-controlled scope experience:**
Mark only one oval per row.

	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
The system was intuitive.					
The system was easy to use.					
I needed support by the test administrator to be able to use the system.					
Most people would quickly learn how to use the system.					
I felt confident using the system.					
I needed more training to confidently use the system.					
At the end of experiment I felt tired.					

16. **Do you have any suggestion for future improvements?**

Powered by
Google Forms

Acronyms

API application programming interface. 6–8, 19

blob binary large object. 24, 34

CD control-display. 14

CPU central processing unit. 7, 15, 28, 35

DOF degree of freedom. 3, 12, 22

EM electromagnetic. 22

FoV field of view. 38

fps frame per second. 12, 23, 38

GPU graphics processing unit. 14, 35

HMD head mounted display. 3

ID index of difficulty. iii, 9, 23, 24, 29, 30, 34

IDE integrated development environment. 18

IMU inertial measurement unit. iii, 2, 11–13, 17–19, 25, 35

IR infrared. 11

JSON JavaScript Object Notation. 20

KITTI KITTI Vision Benchmark Suite. 35

MIS minimally invasive surgery. iii, 2, 4, 10, 34

OpenCV Open Source Computer Vision Library. 3, 6, 15, 19, 35

OR operating room. iii, 11, 33, 34

RMS root mean square. 11

RMSE root mean square error. 22

ROS Robot Operating System. 4, 8, 17, 18, 20, 21, 28, 38, 40

SLERP spherical linear interpolation. 14

UMCG Universitair Medisch Centrum Groningen. iv, 4, 23, 25, 29

UWP Universal Windows Platform. 17, 20

VATS video-assisted thoracoscopic surgery. 1

VR virtual reality. 11

YAML YAML Ain't Markup Language. 9, 38

Bibliography

Agasthian, T. (2013), Video Assisted Thoracoscopic (VATS) Thymectomy, [Online; accessed October 30, 2018].
https://www.ctsnet.org/article/video-assisted-thoracoscopic-vats-thymectomy

Bouguet, J.-Y. (2001), Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm, **vol. 5**, no.1-10, p. 4.

Brox, T., A. Bruhn, N. Papenberg and J. Weickert (2004), High accuracy optical flow estimation based on a theory for warping, in *European conference on computer vision*, Springer, pp. 25–36.

de Bruin, G. (2010), *Endoscope control by head movements applied to minimally invasive surgery*, Master's thesis, University of Twente.

Casiez, G., D. Vogel, R. Balakrishnan and A. Cockburn (2008), The impact of control-display gain on user performance in pointing tasks, **vol. 23**, no.3, pp. 215–250.

Devernay, F. and O. Faugeras (2001), Straight lines have to be straight, **vol. 13**, no.1, pp. 14–24.

Farnebäck, G. (2003), Two-frame motion estimation based on polynomial expansion, in *Scandinavian conference on Image analysis*, Springer, pp. 363–370.

Fischler, M. A. and R. C. Bolles (1981), Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, **vol. 24**, no.6, pp. 381–395.

Friston, S., P. Karlström and A. Steed (2016), The effects of low latency on pointing and steering tasks, **vol. 22**, no.5, pp. 1605–1615.

Geiger, A., P. Lenz, C. Stiller and R. Urtasun (2015), KITTI Optical Flow Evaluation 2015, [Online; accessed October 31, 2018].
http://www.cvlibs.net/datasets/kitti/eval_scene_flow.php?benchmark=flow

Heeger, D. J. (1987), Model for the extraction of image flow, **vol. 4**, no.8, pp. 1455–1471.

Heikkila, J. and O. Silven (1997), A four-step camera calibration procedure with implicit image correction, in *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, IEEE, pp. 1106–1112.

Kalan, S., S. Chauhan, R. F. Coelho, M. A. Orvieto, I. R. Camacho, K. J. Palmer and V. R. Patel (2010), History of robotic surgery, **vol. 4**, no.3, pp. 141–147.

Kannala, J. and S. S. Brandt (2006), A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses, **vol. 28**, no.8, pp. 1335–1340.

Kihara, K., Y. Fujii, H. Masuda, K. Saito, F. Koga, Y. Matsuoka, N. Numao and K. Kojima (2012), New three-dimensional head-mounted display system, TMDU-S-3D system, for minimally invasive surgery application: Procedures for gasless single-port radical nephrectomy, **vol. 19**, no.9, pp. 886–889.

Kopf, J., M. F. Cohen, D. Lischinski and M. Uyttendaele (2007), Joint bilateral upsampling, **vol. 26**, no.3, p. 96.

Lemaignan, S., F. Garcia, A. Jacq and P. Dillenbourg (2016), From Real-time Attention Assessment to "With-me-ness" in Human-Robot Interaction, in *Proceedings of the 2016 ACM/IEEE Human-Robot Interaction Conference*.
http://github.com/severin-lemaignan/gazr

MacKenzie, I. S., A. Sellen and W. A. Buxton (1991), A comparison of input devices in element pointing and dragging tasks, in *Proceedings of the SIGCHI conference on Human factors in*

computing systems, ACM, pp. 161–166.

Martinez, A. M., J. V. Gomez, R. O. Flores and D. L. Espinoza (2009), Postural mechatronic assistant for laparoscopic solo surgery (PMASS), **vol. 23**, no.3, p. 663.

Menze, M. and A. Geiger (2015), Object Scene Flow for Autonomous Vehicles, in *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Mettler, L., M. Ibrahim and W. Jonat (1998), One year of experience working with the aid of a robotic assistant (the voice-controlled optic holder AESOP) in gynaecological endoscopic surgery., **vol. 13**, no.10, pp. 2748–2750.

Noonan, D. P., G. P. Mylonas, J. Shang, C. J. Payne, A. Darzi and G.-Z. Yang (2010), Gaze contingent control for an articulated mechatronic laparoscope, in *Biomedical Robotics and Biomechatronics (BioRob), 2010 3rd IEEE RAS and EMBS International Conference on*, IEEE, pp. 759–764.

Paulich, M., M. Schepers, N. Rudigkeit and G. Bellusci (2018), Xsens MTw Awinda: Miniature Wireless Inertial-Magnetic Motion Tracker for Highly Accurate 3D Kinematic Applications.

Raffa87 (2017), xsense-awinda, [Online; accessed November 16, 2018]. https://github.com/Raffa87/xsense-awinda

Reilink, R. (2013), *Image-based robotic steering of advanced flexible endoscopes and instruments*, Ph.D. thesis.

Reilink, R., G. de Bruin, M. Franken, M. A. Mariani, S. Misra and S. Stramigioli (2010), Endoscopic camera control by head movements for thoracic surgery, in *Biomedical Robotics and Biomechatronics (BioRob), 2010 3rd IEEE RAS and EMBS International Conference on*, IEEE, pp. 510–515.

Romero, A. M. (2014), ROS/Concepts - ROS Wiki, [Online; accessed October 31, 2018]. http://wiki.ros.org/ROS/Concepts

Rozeboom, E. (2016), *Robotic steering of flexible endoscopes*, Ph.D. thesis.

Rozeboom, E. D., J. G. Ruiter, M. Franken, M. P. Schwartz, S. Stramigioli and I. A. Broeders (2014), Single-handed controller reduces the workload of flexible endoscopy, **vol. 8**, no.4, pp. 319–324.

Shoemake, K. (1985), Animating rotation with quaternion curves, in *ACM SIGGRAPH computer graphics*, volume 19, ACM, pp. 245–254.

So, R. H. and M. J. Griffin (2000), Effects of a target movement direction cue on head-tracking performance, **vol. 43**, no.3, pp. 360–376.

Soukoreff, R. W. and I. S. MacKenzie (2004), Towards a standard for pointing device evaluation, perspectives on 27 years of FittsâĂŹ law research in HCI, **vol. 61**, no.6, pp. 751–789.

van der Stap, N., C. H. Slump, I. A. Broeders and F. van der Heijden (2014), Image-based navigation for a robotized flexible endoscope, in *International Workshop on Computer-Assisted and Robotic Endoscopy*, Springer, pp. 77–87.

van der Stap, N., L. Voskuilen, G. de Jong, H. J. Pullens, M. P. Schwartz, I. Broeders and F. van der Heijden (2015), A Real-Time Target Tracking Algorithm for a Robotic Flexible Endoscopy Platform, in *International Workshop on Computer-Assisted and Robotic Endoscopy*, Springer, pp. 81–89.

Tao, M., J. Bai, P. Kohli and S. Paris (2012), SimpleFlow: A Non-iterative, Sublinear Optical Flow Algorithm, in *Computer Graphics Forum*, volume 31, Wiley Online Library, pp. 345–353.

Tomasi, C. and T. Kanade (1991), Detection and tracking of point features.

Van Ranst, W., T. Goedemé and J. Vennekens (2016), Automatic Endoscopic Image Orientation Stabilisation with Ultra-Low-Latency, **vol. 11**, no.2, pp. 119–131.

Wang, V., F. Salim and P. Moskovits (2013), Introduction to HTML5 WebSocket, in *The Definitive Guide to HTML5 WebSocket*, Springer, pp. 1–12.

Whitney, D. (2018), UWP Compatible ROS# Library, [Online; accessed November 12, 2018]. https://github.com/dwhit/ros-sharp

Willow Garage (2018), Robot Operating System - Introduction, [Online; accessed October 31, 2018]. http://wiki.ros.org/ROS/Introduction

YAML.org (2006), YAML Ain't Markup Language, [Online; accessed October 31, 2018]. http://yaml.org/

Zhang, X. and I. S. MacKenzie (2007), Evaluating eye tracking with ISO 9241-part 9, in *International Conference on Human-Computer Interaction*, Springer, pp. 779–788.

Zhang, Z. (2000), A flexible new technique for camera calibration, **vol. 22**, no.11, pp. 1330–1334.